AD-A267 509

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	MAY 1993	~~THESIS~~/DISSERTATION

4. TITLE AND SUBTITLE

"Computational Aerodynamics with Icing Effects"

5. FUNDING NUMBERS

6. AUTHOR(S)

Thomas N. Mouch

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

AFIT Student Attending: University of Kansas

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT/CI/CIA- 93-007D

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

DEPARTMENT OF THE AIR FORCE
AFIT/CI
2950 P STREET
WRIGHT-PATTERSON AFB OH 45433-7765

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for Public Release IAW 190-1
Distribution Unlimited
MICHAEL M. BRICKER, SMSgt, USAF
Chief Administration

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

93 8 05 1 3

93-18100

14. SUBJECT TERMS	15. NUMBER OF PAGES
	156
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. Z39-18
298-102

Abstract

A quick, inexpensive technique has been developed for the analysis of a full aircraft configuration with iced surfaces. A comprehensive literature search of icing analysis methods is presented. Viscous effects for the flow field about an airfoil with an iced leading edge are accounted for in a thin-layer Navier-Stokes code (ARC2D). A panel code (PMARC) solves the flow field away from the body. The results of the airfoil analysis represent the near-field solutions and are used to modify the boundary conditions in the three-dimensional calculations with the panel code by matching the local circulation. This process is repeated until the total lift coefficient between successive iterations differs by less than a specified value. Comparison with viscous experimental data shows excellent results for lift coefficient and a strong improvement over the basic PMARC for drag and pitching moment coefficients. For the full configuration considered, with ice simulated on the horizontal tail, pitching moment data predicts a very sudden unstable pitch break above angle of attack = 8°. This tendency models the pitch tendency described in the literature for a similar configurations with an iced horizontal tail. Thus, a quick method has been developed to handle a full configuration with viscous effects.

Chapter 7

References

1. "Aircraft Icing Avoidance and Protection," NTSB-SR-81-1, National Transportation Safety Board, Washington, D.C., 20594.

2. "Annual Review of Aircraft Accident Data, U.S. General Aviation Calendar Year 1978," NTSB-ARG-80-1, National Transportation Safety Board, Washington, D.C., 10594.

3. "Annual Review of Aircraft Accident Data, U.S. General Aviation Calendar Year 1980," NTSB/ARG-84/01, National Transportation Safety Board, Washington, D.C., 10594.

4. "Annual Review of Aircraft Accident Data, U.S. General Aviation Calendar Year 1984," NTSB/ARG-87/02, National Transportation Safety Board, Washington, D.C., 10594.

5. "Aircraft Accident Report, Ozark Air lines, Inc., DC-9-15, Sioux City Airport, Sioux City, Iowa, December 27, 1968." NTSB-AAR-70-20, Washington, D.C. 20591.

6. "NTSB Seeks Cause in Crash of 737", Aviation Week and Space Technology, January 18, 1982.

7. Excerpts from the NTSB Accident Report on the Air Florida 737 Crash, Aviation Week and Space Technology, October 4, 1982, October 18, 1982 and November 22, 1982.

8. "Crash Investigators Seek Clues in F-28's Major Systems," Aviation Week and Space Technology, March 30, 1992.

9. "Icing Procedures Under Investigation," Flight International, 1-7 April, 1992.

10. Dow, J. P., Sr., Lium, G. D., and Perkins, P. J., "Tailplane Stall, The Rime Is The Reason", NBAA Aviation Issues, Feb, 1993.

11. "Airworthiness Standards: Normal, Utility, Acrobatic, and Commuter Category Airplanes", Code of Federal Regulations, Title 14, Pt. 23, 1991.

12. "Airworthiness Standards: Transport Category Airplanes", Code of Federal Regulations, Title 14, Pt. 25, 1991.

13. Ferarrio, F. and Wallis, T. E., "SF 600 'Kangaroo' Tanker Development and Icing Flight Test Results", 23rd Annual Symposium Proceedings, Society of Flight Test Engineers, August 3-7, 1992.

14. Laschka, B. and Jesse, R. E., "Ice Accretion and its Effects on Aerodynamics of Unprotected Aircraft Components," AGARD Advisory Report 127, Nov. 1978.

15. Reinmann, J.J., Shaw, R.J., and Ranaudo, R.J., "NASA's Program on Icing Research and Technology," NASA TM 101989, Symposium on Flight in Adverse Environmental Conditions, Gol, Norway, May 8-12, 1989.

16. Bragg, M. B. and Gregorek, G. M., "Predicting Aircraft Performance Degradation Due to Ice Accretion," SAE Paper 830742, Business Aircraft Meeting and Exposition, Wichita, KS, April 12 15, 1983.

17. Potapczuk, M. G., Bidwell, C. S., and Berkowitz, B. M., "Progress Toward the Development of an Airfoil Icing Analysis Capability," NASA Computational Fluid Dynamics Conference, Moffett Field, CA, March 7-9, 1989.

18. Potapczuk, M. G. and Bidwell, C. S., "Swept Wing Ice Accretion Modeling," AIAA 90-0756, 28th Aerospace Sciences Meeting, Reno, NV, January 8-11, 1990.

19. Reinman, J. J., "NASA's Aircraft Icing Technology Program," NASA-TM-104518, 1991 Winter Annual Meeting of the ASME, Atlanta, GA, 1-6 Dec 1991.

20. Bragg, M. B., "Experimental Aerodynamic Characteristics of an NACA 0012 airfoil with Simulated Glaze Ice," Journal of Aircraft, Vol. 25, No. 9, September, 1988.

21. Bidwell, C. S., "Icing Characteristic of a Natural-Laminar-Flow, a Medium-Speed, and a Swept, Medium-Speed Airfoil," NASA TM 103693, January, 1991.

22. Bragg, M. B., Gregorek, G. M., and Lee, J. D., "Airfoil Aerodynamics in Icing Conditions," Journal of Aircraft, Vol. 23, No. 1, January 1986.

23. Korkan, K. D.., Cross, E. J., Jr., and Cornell, C.C., "Experimental Aerodynamic Characteristics of an NACA 0012 Airfoil with Simulated Ice," Journal of Aircraft, Vol. 22, No. 2, February, 1985.

24. Flemming, R.J., and Lednicer, D. A., "Correlation of Icing Relationships with Airfoil and Rotorcraft Icing Data," Journal of Aircraft, Vol. 23, No. 10, October 1986.

25. Bragg, M. B., Khodadoust, A., Soltani, R., Wells, S. and Kerho, M., "Aerodynamic Measurements on a Finite Wing with Simulated Ice," AIAA-91-3217-CP, 1991.

26. Ranaudo, R.J., Batterson, J. G., Reehorst, A. L., Bond, T. H., and O'Mara, T. M., "Effects of Horizontal Tail Ice on Longitudinal Aerodynamic Derivatives," Journal of Aircraft, Vol. 28, No. 3, March 1991.

27. Potapczuk, M. G., "Numerical Analysis of an NACA 0012 Airfoil with Leading-Edge Ice Accretions," Journal of Aircraft, Vol. 25, No. 3, March 1988.

28. Cebeci, T., "Calculation of Flow over Iced Airfoils", AIAA Journal, Vol. 27, No. 7, July 1989.

29. Cebeci, T., Chen, H. H., and Alemdaroglu, N., "Fortified LEWICE with Viscous Effects," Journal of Aircraft, Vol. 28, No. 9, Sept. 1991.

30. Ashby, Dale L., Dudley, Michael R., and Iguchi, Steven K. "Development and Validation of an Advanced Low-Order Panel Method." NASA TM-101024, October 1988.

31. Ashby, Dale L., Dudley, Michael R., and Iguchi, Steven K. "Potential Flow Theory and Operation Guide for the Panel Code PMARC," NASA TM-102851, March 1990.

32. Pulliam, T. H., "Euler and Thin-Layer Navier-Stokes Codes: ARC2D, ARC3D," Notes for Computational Fluid Dynamics User's Workshop, The University of Tennessee Space Institute, TN, 1984.

33. Shaw, R.J., Potapczuk, M. G., and Bidwell, C. S., "Predictions of Airfoil Aerodynamic Performance Degradation Due to Icing," NASA TM 101434, Prepwered for the Fourth Symposium on Numerical and Physical Aspects of Aerodynamic Flows, January, 1989.

34. Maskew, Brian. "Program VSAERO Theory Document--a Computer Program for Calculating Nonlinear Aerodynamic Characteristics of Arbitrary Configurations." NASA CR-4023, September 1987.

35. Sorenson, R. L., "A Program to Generate Two-dimensional Grids About Airfoils and Other Shapes by the Use of Poison's Equations," NASA TM 81198, May 1980.

36. Nathman, J. K., and Strash, D. J., "Calculated Longitudinal Stability of an Aircraft With Ice Accumulation On Its Horizontal Stabilizer," Analytical Methods Report 8901, Analytical Methods, Inc., Redmond, WA, August, 1989.

37. Lan, C. E. "Theoretical Prediction of Wing Rocking." Paper No. 32 in AGARD CP-386, presented at the AGARD Symposium on Unsteady Aerodynamics--Fundamentals and Applications to Aircraft Dynamics, May 6-9,1985, Gottingen, FRG.

38. Abbott, I. H., and von Doenhoff, A. E., Theory of Wing Sections, Dover Publications, Inc., New York, 1959.

39. Tseng, J. B., and Lan, C. E. "Calculation of Aerodynamic Characteristics of Airplane Configurations at High Angles of Attack." NASA CR-4182, October 1988.

40. Lan, C. E., Emdad, H., Chin, S.; Sundaram, P., and Mehrotra, S. C., "Calculation of High Angle-of-Attack Aerodynamics of Fighter Configurations." AIAA Paper 89-2188CP, August 1989.

41. Degani, D. and Schiff, L. B., "Computation of Turbulent Supersonic Flows around Pointed Bodies having Crossflow Separation," Journal of Computational Physics, Vol. 66, 1986.

42. Walatka, P. P., Bunig, P. G. Pierce, L. and Elson, P. A., PLOT3D User's Manual, NASA TM 101067, July 1992.

43. Zaman, K. B. M. Q., and Potapczuk, M. G., "The Low Frequency Oscillation in the Flow Over a NACA0012 Airfoil with an 'Iced' Leading Edge", Low Reynolds Number Aerodynamics, T. J. Mueller (Editor), Springer-Verlag Berlin, Heidelberg, 1989.

44. Potapczuk, M. G. Navier-Stokes Analysis of Airfoils with Leading Edge Ice Accretions, PhD. Dissertation, University of Akron, 1989.

45. Bragg, M. B. and Khodadoust, A., "Effect of Simulated Glaze Ice On A Rectangular Wing," AIAA 86-0484, 24th Aerospace Sciences Meeting, Reno, NV, Jan. 1986.

46. Critzos, C. C., Heyson, H. H. and Boswinkle, R. W., Jr., "Aerodynamic Characteristics of NACA 0012 Airfoil Section at Angles of Attack From 0° to 180°", NACA TN 3361, January 1955.

47. Maarsingh, R. A., Labrujere, Th. E. and Smith, J., "Accuracy of Various Wall-Correction Methods for 3D Subsonic Wind-Tunnel

Testing", AGARD CP-429, Naples, Italy, 28 September-1 October 1987.

48. Loftin, L. K., Jr. and Smith, H. A., "Aerodynamic Characteristics of 15 NACA Airfoil Sections at Seven Reynolds Numbers From 0.7 x 10^6 to 9.0 x 10^6", NACA TN 1945, October 1949.

49. Personal Telephone Conversation with D. Ashby. NASA Ames Research Center, April 5, 1993.

50. Kohlman, David L., "Aircraft Icing: Meteorology, Protective Systems, Instrumentation and Certification," University of Kansas 1992 Aerospace Short Courses, October 13-15, 1992, Lawrence, KS.

COMPUTATIONAL AERODYNAMICS

WITH

ICING EFFECTS

by

Thomas N. Mouch
B.S., University of Notre Dame, 1977
M.S., University of Notre Dame, 1981

Submitted to the Department of Aerospace Engineering and the Faculty of the Graduate School of the University of Kansas in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Dissertation Committee:

C. Edward Lan
Dr. C. Edward Lan, Chairman

Dr. Jan Roskam

David R. Downing
Dr. David R. Downing

Saeed Farokhi
Dr. Saeed Farokhi

Francis Thomas
Dr. Francis Thomas

Dissertation Defended: May 1993

Acknowledgments

The author would like to express his sincerest thanks and appreciation to his advisors and the people who have provided him with valuable help and (especially at the trying times) much needed encouragement. He would like to express his heartfelt gratitude to his advisor, Dr. C. Edward Lan, who initially defined this study. During the years working on this study, Dr. Lan has continually provided the needed direction and insight to see this research to completion. Dr. Lan, my hat is off to you.

The author would like to recognize in a special way his family who stood by him through this study and encouraged him to "press on", even in the absences that it forced. Denise, thank for keeping things running day-to-day. I owe you. Megan, Erica and Ben, Daddy's done.

To my parents, thank you for instilling in me the work ethic to see this project through to completion.

To my dear friends in the FRL, thank you for the intellectual challenge you provided and valuable feedback which always kept my feet on the ground.

Space does not allow the author to name each and every person who has made a contribution to this work. Though unnamed,

a big THANK YOU.

"Aviation in itself is not inherently dangerous. But to an even greater degree than the sea, it is terribly unforgiving of an carelessness, incapacity or neglect."

Anon.

Table of Contents

List of Symbols

Regular Symbols		Units
c	chord	ft., in.
c_d	section drag coefficient	
c_l	section lift coefficient	
c_m	section moment coefficient	
FAA	Federal Aviation Administration	
NACA	National Advisory Committee for Aeronautics	
NASA	National Aeronautics and Space Administration	
α	angle of attack	deg, rad
b	wing span	ft, in
S	wing area	ft^2, in^2
AR $(=b^2/S)$	aspect ratio	

ARC2D Symbols		Units
ρ	air density	slug/ft.^{3}
u	x-component of velocity	ft/sec
v	y-component of velocity	ft/sec
e	total energy	
p	Pressure	lbs/ft^2
τ	shear stress	
μ	coefficient of viscosity	

a	speed of sound	ft/sec
l	characteristic length	ft
Re	Reynolds number	
Pr	Prandtl number	
ξ	general curvilinear coordinate (along the body)	
η	general curvilinear coordinate (normal to the body)	
∂	partial derivative symbol (usually a subscript to show which coordinate the partial derivative is taken with respect to)	
γ	ratio of specific heats	
ϑ, φ	parameter used to choose differencing scheme	
t	time	sec
Δ	difference between two time steps	
$F(y)$ $(= y\lvert\omega\rvert)$	moment of vorticity	

ARC2D Subscripts

l	laminar
t	turbulent

PMARC Symbols		Units
Φ	total potential	
dS	elemental area	1/ft.^{2}
$\nabla\left(=\frac{\partial}{\partial x}i+\frac{\partial}{\partial y}j+\frac{\partial}{\partial z}k\right)$	differential operator	
r	distance	ft
ϕ	perturbation potential	
μ	doublet strength	
σ	source strength	
V	velocity	ft/sec
B	source influence coefficient matrix	
C	doublet influence coefficient matrix	

GRAPE Symbols		Units
x, y	coordinates in physical space	
ξ,η	coordinates in computational space	
J	Jacobian	
P,Q	inhomogeneous terms of Poisson's equation	

Present Study Symbols		Units
f	a scaling factor	

$c_{\ell(3-D)}$	section lift coefficient from 3-dimensional computations
c_{ℓ_α}	2-dimensional lift curve slope
$c_{\ell(2-D)}$	2-dimensional section lift coefficient
Δ	an increment

Present Study

Subscripts

e	effective
n	geometric
i	induced
o	zero lift

List of Figures

MEMO FOR RECORD

May 19, 1993

The following information is provided as required by AFITR 53-1, paragraph 7-7.

Author:	Thomas N. Mouch
Title:	"Computational Aerodynamics with Icing Effects"
Rank, Service:	Major, USAF
Date Completed	1993
	156 Pages
Degree Awarded:	PhD.
School:	University of Kansas

Chapter 1
Introduction

The effect of airframe icing on aircraft handling qualities and performance is a concern for the majority of aircraft operators. According to the National Transportation Safety Board: "Aircraft structural icing is primarily a problem for the smaller commuter, air taxi and general aviation aircraft. ... Of the approximately 210,000 general aviation aircraft registered in the United States [in 1981], only about 12,000 have been issued certificates by the FAA for flight into known icing conditions."[1] Yet pilots encounter airframe icing unexpectedly and icing is implicated in many accidents each year. From the yearly reviews of aircraft accident data, a significant percentage of the general aviation aircraft accidents due to weather cite "icing conditions" as a cause or factor.[2-4]

Airframe icing is not just a concern for operators of "small" or "light" aircraft: there are many instances of loss of control or fatal crashes of transport aircraft due to icing. (References 5 through 9 provide a short list representative of the many air transport accidents due to icing.) These accident investigations provide data showing the effects of icing on the aircraft performance and handling qualities through the data recorders on the incident aircraft. The amount of ice contamination necessary to significantly affect the airflow over a control or lifting surface is very small. The Fokker F.28, e.g., encounters a "25% reduction in maximum lift and a 6° lower stall angle of attack. The test, with contamination equivalent to ice particles 1-2 mm in diameter about one particle/cm, replicated a 1969 incident…"[8]

As recently as October 1992, the FAA, in an article to airman, discussed the "number of fatal and non-fatal accidents and incidents of uncommanded pitch-down resulting from tailplane stall during or following flight in icing conditions."[10] This article states: "Ice induced tailplane stall accidents of record have occurred with reported ice accretion from 3/16 to 1 inch thick on the leading edge of the tailplane"[10]. The most insidious part of icing phenomena is that "Tailplane ice without wing ice is possible"[10] due to the different physical layouts of different airplanes, i.e. tail immersed in the propwash, differences in leading edge radii of the wing and tail, etc. The article concludes by stating: "Pilot action resulting from proper training using appropriate information can have an immediate benefit in minimizing the hazards of ice induced tailplane stall."[10] Pilots could gain experience flying with an ice accumulation through simulator training. The aircraft's stability and control characteristics could be developed for input to the simulator if a computer program existed to model a complete configuration with ice accumulations. This need is also suggested in Reference 14 and discussed later in this section.

Typically, icing effects are greatest when the aircraft is flying at an angle of attack different from the one at which the ice was accreted. Usually the ice is accreted at a fairly low angle of attack, e.g., during cruise or descent. Then the aircraft flies at a higher angle of attack, e.g., on approach or during a go-around or a subsequent takeoff. The built-up airframe ice causes flow separation from the unprotected or poorly protected lifting or control surfaces. This can lead to aircraft pitchup and stall or uncontrolled lateral-directional motion. The ability to predict this motion quickly would greatly aid in the design of the control system.

The Federal Aviation Regulations[11,12] (F.A.R.'s Pt. 23 & 25) require that "An analysis must be performed to establish ... the adequacy of the ice protection system for the various components of the airplane." Ferrario and Wallis[13] give excellent insight into the workings of an icing flight test program. Through this report, one can understand the time consuming and dangerous nature of certifying an aircraft for flight into known icing conditions. The flight test program involves testing the aircraft with simulated ice shapes, with system failures and flight in actual icing conditions. The currently unpredictable nature of this testing is a major contributor to the hazardous nature of this testing. A tool to conduct some of this analysis at minimum expense could shrink the flight test matrix and reduce program cost.

What is the physical phenomena related to ice formation on the leading edge of a wing? Two general types of ice form on the wing surface based on the rate at which the supercooled water droplets freeze. At colder temperatures, approximately -8°C and below, a dry growth (rime) ice forms and conforms to the shape of the leading edge. Due to the streamline nature of this ice formation, it does "not adversely affect lift and drag characteristics of the airfoil."[14] At temperatures just below freezing and at a higher liquid water content, glaze ice forms. "At these temperatures, water droplets do not freeze immediately after impingement, but run first on the profile upward or downward from the stagnation point before freezing.... The result is a deposit of ice with two horn-like protrusions...Such an ice shape is aerodynamically very unfavourable."[14] Other factors affecting the shape of ice formed include: airspeed, drop size and liquid water content of the cloud, shape of the surface and angle of attack.

Leading edge icing greatly distorts the shape of an airfoil. Figure 1, from icing tunnel experiments, shows the dependence of ice shape on temperature and liquid water content (LWC). Though there is a dependence on temperature and LWC, the mass of the ice shape and the general form of the shape are reasonably constant away from the freezing level.[15] Thus, modeling one ice shape can cover a range of temperatures with little loss of generality.

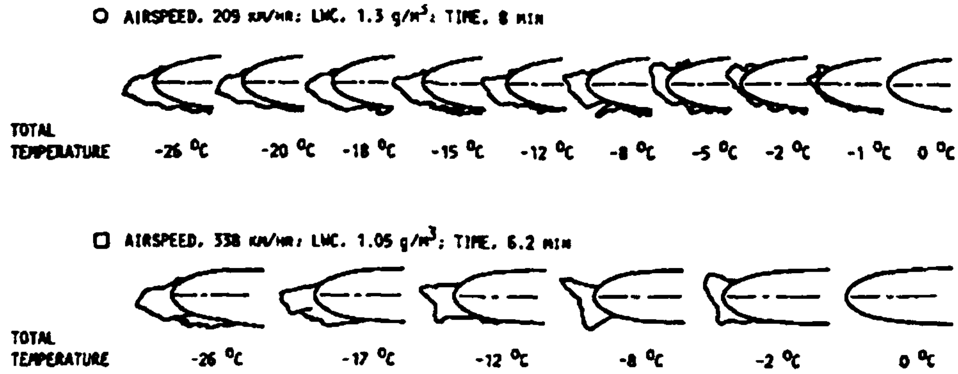

Figure 1. Ice Shapes

(Copied from Reference 15)

Due to the physical aspects of flow over an iced airfoil, a viscous analysis of the flowfield is necessary. Figure 2 shows the pertinent aspects of this flowfield. These key aspects include: the ice shape, or "horn", a separated flow zone and a thick distorted boundary layer. The horn formation, with its accompanying surface roughness, varies due to the type of ice formed. The type of ice formed is based mainly on temperature of the air and the airfoil surface. Behind the "horn" is a separated flow zone which can occur on either the upper or lower surface depending on two factors:

(1) the angle of attack at which the ice was formed and

(2) the angle of attack at which the aircraft is now flying.

Finally, there is a thick distorted boundary layer which alters the airfoil shape further as well as increases the drag.

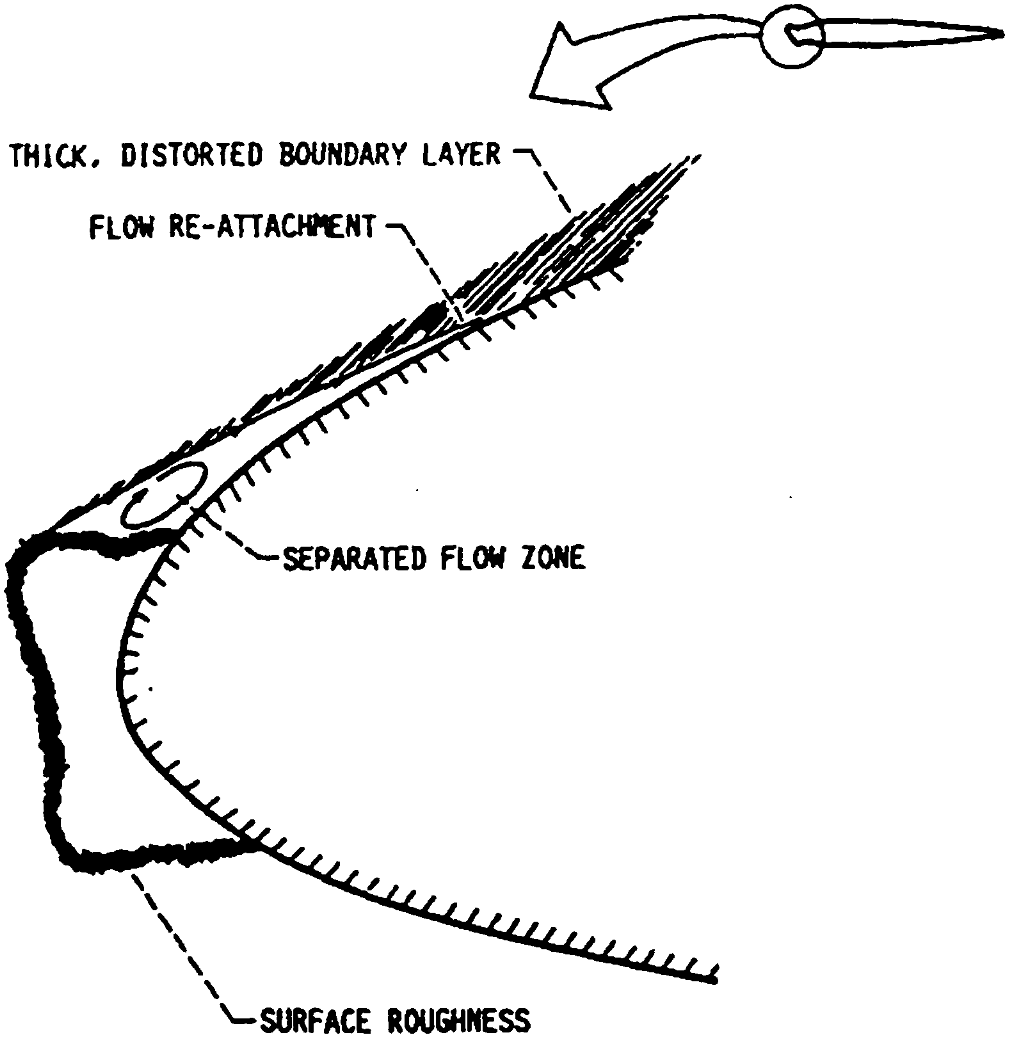

Figure 2. Key Aspects of Airfoil and Wing Icing

(Copied From Reference 15)

Current analysis methods dealing just with an airfoil are not adequate to satisfy the requirements of the F.A.R.'s regarding certification of an aircraft. What is needed is the capability to model the whole aircraft. Bragg and Gregorek[16] provide an analytical scheme using the component buildup method. Limitations to this scheme include:

(1) using empirical correlations which "must come from wind tunnel or flight tests"[16] (Thus the results are very model dependent.);

(2) dealing only with the lift and drag of the iced aircraft and not the moments that affect the handling qualities.

Potapczuk et al.[17] provide a step toward three-dimensional ice accretion by modeling the ice accretion for a swept wing. This is a first step toward modeling the accretion of ice on a complete aircraft, but the results don't compare well with the 2-D data. Currently attempts are being made to model the entire iced aircraft in a viscous medium, but are meeting with limited success.[18] Thus analysis of a complete aircraft with icing is not available yet.

Three-dimensional computation with the full Navier-Stokes equations for an entire aircraft is the ultimate solution due to the separated flows involved. These calculations would be very time consuming both in generating the grid for the flow solver and in the computing time required for the solution. A quick, less expensive solution involves the use of a panel code, like PMARC (**P**anel **M**ethod, **A**mes **R**esearch **C**enter), to model the entire aircraft. But potential flow methods cannot model the rotational flow in the separated wake behind the ice shape. Therefore, the proposed approach is to use a Navier-Stokes code (ARC2D) to provide the near field solutions for a panel code (PMARC) as described under Theoretical Development.

Such a tool would meet one of three key objectives of NASA's Icing Technology Program: "numerically simulate an aircraft's response to an inflight icing encounter."[19]

Chapter 2
Review of Literature

This section will provide a review of pertinent literature. Experimental testing will be discussed first, followed by a discussion of analytical results.

2.1 Experimental

Much experimental work has been reported on the effect of ice accreted to the leading edge of airfoils and wings. Bragg and Gregorek[19] used the "component buildup" method to estimate the effect of ice accretion on the performance of aircraft. In their method, they used existing component data and empirical correlations (or experimental 2-D data) to predict aircraft performance with ice. This method relies on a large database developed from experimental testing including flights into icing encounters.

To develop this part of the database, NASA Lewis Research Center has a Icing Research Tunnel (IRT) which simulates the effects conducive to airframe icing to determine the effects on airfoil performance. Bidwell[21], as one example, used the IRT to determine icing characteristics (shape and drag increase) of three different airfoils. These shapes can be used to model ice accretion. Because of the effects of icing tunnel conditions on experimental equipment, most flowfield measurements for an iced airfoil have been made in a "clean" tunnel using a simulated ice shape which closely models the shapes developed in the icing tunnel. This is exactly what Bragg[20] and Bragg, et al.[22] did to conduct their investigations. Bragg, et al., used ice tracings to duplicate the ice shapes and drag measurements to verify their methods. (Drag is the only value which can be directly measured during ice accretion in the IRT.) Bragg used these shapes attached to the leading edge of a NACA 0012 to take pressure, lift, drag and moment measurements at a variety of angles of attack and Mach numbers. The effect of an iced shape on c_l, c_d, and $c_{m_{c/4}}$ is compared with the data for a "clean" (no ice shape attached) NACA 0012 from Abbot and von Doenhoff[38] in Figure 3.

In these figures, one can see the reduction in both lift curve slope and $c_{l_{max}}$ for the iced airfoil. These reductions along with the large increase in c_d are attributed to the massive separation region behind the "horns" of the iced shape. Notice the large increase in drag coefficient and large change in pitching moment due to the distorted pressure distribution behind the iced shape.

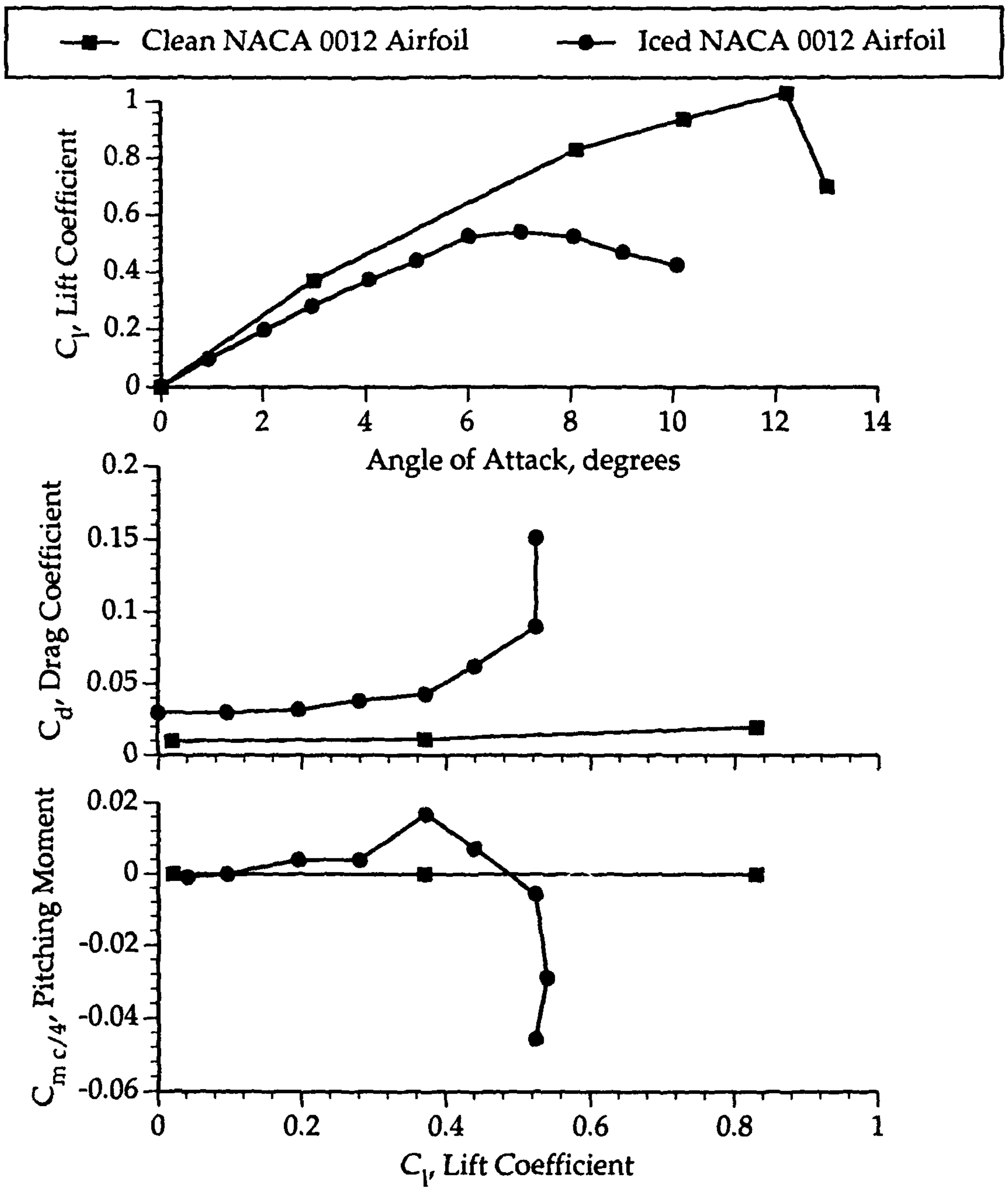

Figure 3. Effect of Iced Shape on c_l, c_d, $c_{m_{c/4}}$ for a NACA 0012 Airfoil

(Clean Airfoil Data from Ref. 38, Iced Airfoil Data from Ref. 20.)

Korkan[23] extended this work to take into account Reynolds number effects in terms of Δc_d , Δc_l, $\Delta c_{l_{max}}$, and $\Delta c_{m_{c/4}}$ due to ice accretion for a generic ice shape. Flemming and Lednicer[24] moved the experimental database for both accreted ice and simulated ice into the high-speed regime, considering Mach numbers up to 0.7. These results improve the prediction capability for ice thickness, lift, drag and pitching moment by increasing the number of parameters considered to predict the type of ice formed. The method breaks down in the approach to stall regime.

Bragg, et al.[25] extended simulated icing force measurements in the third dimension by considering both straight and swept wings with simulated ice. Flow visualization from this study shows the three-dimensional nature of the flowfield about the iced swept wing.

To meet certification requirements, much full-scale testing of icing effects on aircraft has been accomplished, but most of this data is proprietary. Laschka and Jesse[14] discuss the decision-making process and testing required to certify the Airbus A300 without de-icing or anti-icing protection on the tail surfaces. This investigation showed that tail icing did have an effect on the handling qualities of the aircraft, but “not too serious”[14] an effect.

Ranaudo, et al.[26] performed tests to determine the accuracy with which the effects of icing could be measured for aircraft longitudinal stability and control. In this study they stated that even if the aircraft is equipped with energy-efficient deicing systems, they “...will have to demonstrate acceptable flying qualities with some leading edge contamination..."[26] From this study, it was determined that the main effect of icing is at low speed.

2.2 Theoretical

There are two favored methods to perform the viscous analysis of an airfoil with a leading edge ice shape attached: the interactive boundary-layer method and *solution of the Navier-Stokes equations.*

Cebeci[28] uses the interactive boundary-layer(IBL) method to calculate the viscous effects of an iced airfoil. In this method, an inviscid conformal mapping solution is made about the airfoil. Then, the boundary layer equations are solved for this pressure distribution. This boundary-layer solution modifies the airfoil shape and the cycle is repeated. Cebeci[29], et al., have applied this technique to calculate the forces and moments for an airfoil with an ice accretion shape developed by LEWICE. (LEWICE is a computer program developed at NASA Lewis Research Center to simulate the accretion of ice on an airfoil.) This method requires some "adjustment" to the iced shape to smoothly introduce the airfoil with this iced shape into the calculations.

Potapczuk[27] used ARC2D (Ames Research Center, Two (2)-Dimensional), a thin-layer Navier-Stokes code, to calculate the forces and moments on a NACA 0012 airfoil with leading edge ice. The NACA 0012 model was modified to have a leading edge ice shape that had the gross cross sectional features of an ice shape grown in the IRT, but also had a geometry that could be accurately digitized to allow input to the flow analysis codes. This study states: "Computational results agree well with experimental information at angles of attack below stall."[27]

Reinmann et al.[15], in a NASA summary paper on icing research, indicate that both methods predict lift and drag coefficients well at low angles of attack (a < 6°), but "the IBL code appeared inadequate at the high alphas. At these higher angles of attack the ARC2D code predicted unsteady flow." This report aided in the decision to use ARC2D to provide the viscous analysis.

Chapter 3
Theoretical Development

The primary objective in this research is to compute the longitudinal aerodynamics for a complete airplane configuration with iced lifting surfaces. Right now, there are no simple and inexpensive methods that can deal successfully with this problem. This section provides a theoretical development of the programs to be used in this research. These programs include: ARC2D, a viscous flow analysis code, PMARC, an inviscid panel code and GRAPE, a grid generation code.

The proposed method is to use a panel code (PMARC)[30,31] to model the complete configuration, while the icing effect on a lifting surface is to be calculated with a two-dimensional Navier-Stokes code (ARC2D)[32]. How these two solutions are utilized to obtain the final results for a complete airplane configuration represents the important contribution of this research.

3.1 ARC2D

The viscous effects on an iced airfoil will be precomputed using ARC2D and stored in a "lookup file" to be referenced by PMARC. ARC2D is based on the thin-layer Navier-Stokes equations in two-dimensions[32].

The strong conservation form of the two-dimensional Navier-Stokes equations in Cartesian coordinates and nondimensional form can be written as follows:

$$\partial_t Q + \partial_x E + \partial_y F = \mathrm{Re}^{-1}\left(\partial_x E_v + \partial_y F_v\right) \tag{3.1}$$

where:

$$Q = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ e \end{bmatrix}, E = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ u(e+p) \end{bmatrix}, F = \begin{bmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ v(e+p) \end{bmatrix} \tag{3.2a}$$

$$E_v = \begin{bmatrix} 0 \\ \tau_{xx} \\ \tau_{xy} \\ f_4 \end{bmatrix}, F_v = \begin{bmatrix} 0 \\ \tau_{xy} \\ \tau_{yy} \\ g_4 \end{bmatrix} \tag{3.2b}$$

with

$$\begin{aligned} \tau_{xx} &= \mu(4u_x - 2v_y)/3 \\ \tau_{xy} &= \mu(u_x - v_y) \\ \tau_{yy} &= \mu(-2u_x + 4v_y)/3 \\ f_4 &= u\tau_{xx} + v\tau_{xy} + \mu \Pr^{-1}(\gamma-1)^{-1}\partial_x a^2 \\ g_4 &= u\tau_{xy} + v\tau_{yy} + \mu \Pr^{-1}(\gamma-1)^{-1}\partial_y a^2 \end{aligned} \tag{3.2c}$$

The equation of state relates the flow variables, Q, to pressure:

$$p = (\gamma - 1)\left(e - \frac{1}{2}\rho\left(u^2 + v^2\right)\right) \tag{3.3}$$

In ARC2D the primary variables, ρ (density), u v (Cartesian velocities), and e (total energy), are non-dimensionalized as:

$$\tilde{\rho} = \frac{\rho}{\rho_\infty}, \tilde{u} = \frac{u}{a_\infty}, \tilde{v} = \frac{v}{a_\infty}, \tilde{e} = \frac{e}{\rho_\infty a_\infty^2} \tag{3.4a}$$

Time is nondimensionalized as $\hat{t} = {}^{ta_\infty}\!/\!_{l}$, where l is a characteristic length.

The viscous coefficients are non-dimensionalized as:

$$\tilde{\mu} = \frac{\mu}{\mu_\infty}, \mathrm{Re} = \frac{\rho_\infty l a_\infty}{\mu_\infty} \tag{3.4b}$$

Note that R_e uses a_∞, therefore R_e, based on u_∞ (the usual form for experimental data), must be scaled by $M_\infty = u_\infty / a_\infty$. For the remainder of the development, the ~ will be dropped for simplicity.

Equation (3.1) can be transformed from Cartesian coordinates to general curvilinear coordinates where

$$\tau = t \qquad \xi = \xi(x,y,t) \qquad \eta = \eta(x,y,t) \tag{3.5}$$

The transformations are chosen so that the grid spacing in the curvilinear space is uniform and of unit length. Chain rule expansions are used to represent the Cartesian derivatives ∂_x and ∂_y of Eq. (3.1) in terms of the curvilinear derivatives:

$$\begin{bmatrix} \partial_t \\ \partial_x \\ \partial_y \end{bmatrix} = \begin{bmatrix} 1 & \xi_t & \eta_t \\ 0 & \xi_x & \eta_x \\ 0 & \xi_y & \eta_y \end{bmatrix} \begin{bmatrix} \partial_\tau \\ \partial_\xi \\ \partial_\eta \end{bmatrix} \tag{3.6}$$

Therefore, applying Eq. (3.6) to the Navier-Stokes Equations, Eq (3.1), one gets

$$\begin{aligned} &\partial_\tau Q + \xi_t \partial_\xi Q + \eta_t \partial_\eta Q + \xi_x \partial_\xi E + \eta_x \partial_\eta E + \xi_y \partial_\xi F + \eta_y \partial_\eta F = \\ &\mathrm{Re}^{-1}\left(\xi_x \partial_\xi E_v + \eta_x \partial_\eta E_v + \xi_y \partial_\xi F_v + \eta_y \partial_\eta F_v\right) \end{aligned} \tag{3.7}$$

To this equation in generalized curvilinear coordinates, one can apply the thin-layer approximation. This approximation requires that:

(1). All body surfaces be mapped onto coordinate surfaces

(2). Grid spacing is clustered to the body surfaces such that sufficient resolution for a particular Reynolds number is obtained.

(3). All the viscous derivatives in the ξ-direction are neglected, while the terms in the η-direction are retained. All of the inviscid terms are used.

After applying this approximation to Eq. (3.7), one obtains:

$$\partial_\tau \hat{Q} + \partial_\xi \hat{E} + \partial_\eta \hat{F} = \mathrm{Re}^{-1} \partial_\eta \hat{S} \tag{3.8}$$

where

$$\hat{Q} = J^{-1}\begin{bmatrix} \rho \\ \rho u \\ \rho v \\ e \end{bmatrix}, \hat{E} = J^{-1}\begin{bmatrix} \rho U \\ \rho u U + \xi_x p \\ \rho v U + \xi_y p \\ U(e+p) - \xi_t p \end{bmatrix}, \hat{F} = J^{-1}\begin{bmatrix} \rho V \\ \rho u V + \eta_x p \\ \rho v V + \eta_y p \\ V(e+p) - \eta_t p \end{bmatrix} \tag{3.9a}$$

with

$$U = \xi_t + \xi_x u + \xi_y v, \quad V = \eta_t + \eta_x u + \eta_y v \tag{3.9b}$$

and

$$\hat{S} = J^{-1}\begin{bmatrix} 0 \\ \eta_x m_1 + \eta_y m_2 \\ \eta_x m_2 + \eta_y m_3 \\ \eta_x(u m_1 + v m_2 + m_4) + \eta_y(u m_2 + v m_3 + m_5) \end{bmatrix} \tag{3.9c}$$

and

$$\begin{aligned} m_1 &= \mu(4\eta_x u_\eta - 2\eta_y v_\eta)/3 \\ m_2 &= \mu(\eta_y u_\eta + \eta_x v_\eta) \\ m_3 &= \mu(-2\eta_x u_\eta + 4\eta_y v_\eta)/3 \\ m_4 &= \mu \,\mathrm{Pr}^{-1}(\gamma - 1)^{-1}\eta_x \partial_\eta (a^2) \\ m_5 &= \mu \,\mathrm{Pr}^{-1}(\gamma - 1)^{-1}\eta_y \partial_\eta (a^2) \end{aligned} \tag{3.9d}$$

ARC2D uses the Baldwin-Lomax turbulence model which was specifically designed for use with the thin-layer approximation.[32] This model is appropriate to attached and mildly separated boundary layers. Other turbulence models have been applied to the icing problem. In the study by Shaw, et al.[33], the Johnson-King and the k-ε models were tested with no noticeable effect. This study does conclude that "Limitations ... include ... turbulence modeling."[33] Potapczuk[27] used ARC2D with this turbulence model and had good success in modeling the simulated ice shape to α=7°.

ARC2D uses an implicit approximate factorization finite difference scheme which can be either first or second order accurate in time. Local time linearizations are applied to the nonlinear terms and an approximate factorization of the two-dimensional implicit operator is used to produce locally one-dimensional operators. Approximate factorization is introduced because integration of the full two-dimensional operator is "too expensive."[32] The spatial derivative terms are approximated with second order central differences. Explicit and implicit artificial dissipation terms are added to achieve nonlinear stability. A spatially variable time step is used to accelerate convergence for steady-state calculations.[32]

One arrives at the approximate factored form of the Eq (3.8) by applying an implicit three point finite difference scheme of the form[32]:

$$\Delta\hat{Q}^n = \frac{\vartheta\Delta t}{1+\varphi}\frac{\partial}{\partial t}(\Delta\hat{Q}^n) + \frac{\Delta t}{1+\varphi}\frac{\partial}{\partial t}\hat{Q}^n + \frac{\varphi}{1+\varphi}\frac{\partial}{\partial t}\hat{Q}^{n-1} + O\left[(\vartheta - \frac{1}{2} - \varphi)\Delta t^2 + \Delta t^3\right] \tag{3.10}$$

where:

$$\Delta\hat{Q}^n = \hat{Q}^{n+1} - \hat{Q}^n \tag{3.11a}$$

and

$$\hat{Q}^n = \hat{Q}(n\Delta t) \tag{3.11b}$$

The parameters ϑ and φ can be chosen to produce different schemes of either first or second order accuracy in time. If $\vartheta = 1$ and $\varphi = 1/2$, this scheme is second order in time.

Therefore applying Eq. (3.10) to Eq. (3.8) results in:

$$\hat{Q}^{n+1} - \hat{Q}^n + h(\hat{E}_\xi^{n+1} + \hat{F}_\eta^{n+1} - \mathrm{Re}^{-1}\hat{S}_\eta^{n+1}) = 0 \tag{3.12}$$

with $h = \Delta t$.

We wish to solve Eq. (3.12) for $\hat{Q}^{n+1}$, given $\hat{Q}^n$. The flux vectors $\hat{E}, \hat{F}$ and $\hat{S}$ are nonlinear functions of $\hat{Q}$ and therefore, Eq. (3.12) is nonlinear in $\hat{Q}^{n+1}$. The nonlinear terms are linearized in time about $\hat{Q}^n$ by a Taylor series such that

$$\begin{aligned} \hat{E}^{n+1} &= \hat{E}^n + \hat{A}^n \Delta \hat{Q}^n + O(h^2) \\ \hat{F}^{n+1} &= \hat{F}^n + \hat{B}^n \Delta \hat{Q}^n + O(h^2) \\ \mathrm{Re}^{-1} \hat{S}^{n+1} &= \mathrm{Re}^{-1}\left[\hat{S}^n + J^{-1}\hat{M}^n \Delta \hat{Q}^n\right] + O(h^2) \end{aligned} \tag{3.13}$$

where $\hat{A} = \partial\hat{E}/\partial\hat{Q}$, $\hat{B} = \partial\hat{F}/\partial\hat{Q}$, and $\hat{M} = \partial\hat{S}/\partial\hat{Q}$ are the flux Jacobians and $\Delta\hat{Q}^n$ is $O(h)$.

Applying Eq. (3.13) to Eq. (3.12) and combining the $\Delta\hat{Q}^n$ terms produces the "delta" form of the algorithm

$$\begin{aligned} &\left[I + h\partial_\xi \hat{A}^n + h\partial_\eta \hat{B}^n - \mathrm{Re}^{-1} hJ^{-1}\partial_\eta \hat{M}\right]\Delta\hat{Q}^n = \\ &-h\left(\partial_\xi \hat{E}^n + \partial_\eta \hat{F}^n - \mathrm{Re}^{-1}\partial_\eta \hat{S}^n\right) \end{aligned} \tag{3.14}$$

This is the unfactored form of the algorithm. The right hand side of Eq. (3.14) is called the "explicit" part and the left hand side the "implicit" part. The implicit part can be factored into two one-dimensional operators as

$$\begin{aligned} &\left[I + h\partial_\xi \hat{A}^n + h\partial_\eta \hat{B}^n - \mathrm{Re}^{-1} hJ^{-1}\partial_\eta \hat{M}\right]\Delta\hat{Q}^n = \\ &\underbrace{\left[\mathrm{I} + \mathrm{h}\partial_\xi \hat{\mathrm{A}}^n\right]}_{\xi\text{-direction term}}\underbrace{\left[\mathrm{I} + \mathrm{h}\partial_n \hat{\mathrm{B}}^n - \mathrm{h}\,\mathrm{Re}^{-1}\,\partial_\eta \mathrm{J}^{-1}\hat{\mathrm{M}}^n\right]}_{\eta\text{-dirction term}}\Delta\hat{\mathrm{Q}}^n \\ &\underbrace{-\mathrm{h}^2\partial_\xi \hat{\mathrm{A}}^n \partial_\eta \hat{\mathrm{B}}^n \Delta\hat{\mathrm{Q}}^n - \mathrm{h}^2\,\mathrm{Re}^{-1}\,\partial_\xi \hat{\mathrm{A}}^n \partial_\eta \mathrm{J}^{-1}\hat{\mathrm{M}}^n \Delta\hat{\mathrm{Q}}^n}_{\text{Cross Term}} \end{aligned} \tag{3.15}$$

The cross term is second order accurate since $\Delta\hat{Q}^n$ is $O(h)$. It can therefore be neglected without degrading the time accuracy of any second order scheme which is chosen.

The resulting approximate factored form of the algorithm is

$$\left[I + h\partial_\xi \hat{A}^n\right]\left[I + h\partial_\eta \hat{B}^n - h\,\mathrm{Re}^{-1}\partial_\eta J^{-1}\hat{M}^n\right]\Delta\hat{Q}^n = \\ -h\left(\partial_\xi \hat{E}^n + \partial_\eta \hat{F}^n - \mathrm{Re}^{-1}\partial_\eta \hat{S}^n\right) \tag{3.16}$$

And this allows the solution to consist of two one-dimensional sweeps, one in the ξ–direction and one in the η–direction. The resulting process is much more economical than the unfactored algorithm in terms of computer storage and CPU time.

3.2 PMARC

PMARC (**P**anel **M**ethod **A**mes **R**esearch **C**enter[30,31]) is the panel code which will be used to model the region away from the airplane. This code was derived from VSAERO[34] by including various improvements. These improvements[31] are summarized as follows:

(1). In the PMARC code, the wake generation and relaxation schemes used in the VSAERO code are replaced with a time-stepping wake model. This allows the user to specify a prescribed motion for the paneled geometry.

(2). The time-stepping routines allow either unsteady or steady motions to be prescribed. Also, the time-stepping wake makes it possible to compute aerodynamic data for complete aircraft configurations going through maneuvers.

(3). "A data management scheme has been devised for PMARC which seeks to maximize the number of panels the code can handle while minimizing the amount of memory and disk scratch space required to run the code."[31] Specific aspects of the data management scheme include use of variable dimensioning for all major arrays within the code, creation of a memory saving common block in which to store arrays local to a subroutine, provision of a reasonable balance between the amount of memory used and the amount of disk scratch space used, and elimination of redundancy of variables both within the code and the plot and output files.

(4). The PMARC code has adjustable arrays for all the geometry, wake, and solution-related arrays. This allows the user to customize the size of the code to fit his particular needs and hardware capacity.

(5). In developing PMARC, many of the disk scratch files, that VSAERO used, were removed to conserve memory. Removal of the disk IO statements and use of common blocks to pass information between subroutines greatly streamlines the coding and produces a faster running code.

(6). The aerodynamic data section of the output file from PMARC has been reorganized to add new options to the panel aerodynamic data printout and to separate the force and moment data from the panel aerodynamic data.

In PMARC, the flow field is assumed to be inviscid, irrotational, and incompressible. (The following development closely follows the development of Ashby[31].) The body is modeled as a closed surface that divides the flow field into two regions, an external and an internal region. A velocity potential is assumed to satisfy the Laplace equation where:

$$\nabla^2\Phi = 0, \qquad (3.17)$$

applies in the external region and

$$\nabla^2\Phi_i = 0, \tag{3.18}$$

applies in the internal region.

By applying Green's Theorem, the potential at any point, P, may be evaluated:

$$\Phi_P = \frac{1}{4\pi}\iint_{S+W+S_\infty}(\Phi - \Phi_i)\hat{n}\cdot\nabla\left(\frac{1}{\bar{r}}\right)dS - \frac{1}{4\pi}\iint_{S+W+S_\infty}\left(\frac{1}{\bar{r}}\right)\hat{n}\cdot(\nabla\Phi - \nabla\Phi_i)dS \tag{3.19}$$

where $\bar{r}$ is the distance from the point P to the element dS on the surface and $\hat{n}$ is the unit normal to the surface pointing into the flow field of interest. Physically, the first integral represents the disturbance potential from a surface distribution of doublets and the second integral represents the contribution from a surface distribution of sources.

In the development of PMARC, this *equation* is simplified through the following assumptions:

(1). On the surface at infinity, the perturbation potential due the body is zero, leaving only the uniform onset flow.

(2). The wake is thin and there is no entrainment so the source term for the wake is zero and the jump in normal velocity across the wake is zero.

Applying these assumptions, the simplified equation is:

$$\begin{aligned}\Phi_P &= \frac{1}{4\pi}\iint_S(\Phi - \Phi_i)\hat{n}\cdot\nabla\left(\frac{1}{\bar{r}}\right)dS - \frac{1}{4\pi}\iint_S\left(\frac{1}{\bar{r}}\right)\hat{n}\cdot(\nabla\Phi - \nabla\Phi_i)dS \\ &+ \frac{1}{4\pi}\iint_W(\Phi_U - \Phi_L)\hat{n}\cdot\nabla\left(\frac{1}{\bar{r}}\right)dS + \phi_{\infty_P}\end{aligned} \tag{3.20}$$

When performing the integration, the point P must be excluded if it lies on the surface. Assuming a hemispherical deformation of the surface and evaluating the integral as the radius of the hemisphere goes to zero gives a contribution at the point P equaling $\pm \frac{1}{2}(\Phi - \Phi_i)_P$. (The plus sign applies for points lying on the inside of the surface and the minus sign applies for points on the outside of the surface.)

The total potential, Φ, can be viewed as being made up of an onset potential, ϕ_∞, and a perturbation potential, $\phi = \Phi - \phi_\infty$. The potential of the flow internal to the surface is set equal to the onset potential ϕ_∞. (This boundary condition is chosen to reduce the magnitudes of the singularities on the surface.) Using this internal Dirichlet boundary condition and looking at points P inside the surface, Eq.(3.20) can be rewritten as:

$$\begin{aligned}\Phi_P = \frac{1}{4\pi}\iint_{S-P} \phi \hat{n}\cdot\nabla\left(\frac{1}{\bar{r}}\right) dS - \frac{1}{4\pi}\iint_{S}\left(\frac{1}{\bar{r}}\right)\hat{n}\cdot(\nabla\Phi - \nabla\phi_\infty)\, dS \\ + \frac{1}{4\pi}\iint_{W}(\Phi_U - \Phi_L)\hat{n}\cdot\nabla\left(\frac{1}{\bar{r}}\right) dS - \frac{1}{2}\phi_P\end{aligned} \tag{3.21}$$

If one refers to the physical definitions made for Eq. (3.19), the following equations may be written for the doublet and source strengths:

$$4\pi\mu = \phi = \Phi - \phi_\infty \tag{3.22}$$

$$4\pi\sigma = -\hat{n}\cdot(\nabla\Phi - \nabla\phi_\infty) \tag{3.23}$$

Assuming that the normal velocity at the surface is either zero (a solid surface) or some known value (a porous surface for suction or blowing), then the source strengths can be evaluated immediately:

$$\sigma = \frac{1}{4\pi}(V_{norm} - \hat{n}\cdot\bar{V}_\infty) \tag{3.24}$$

Substituting Eqs. (3.22) and (3.23) into Eq. (3.21), leaves the integral equation to be solved for the unknown doublet strength over the surface:

$$0=\left[\iint_{S-P}\mu\hat{n}\cdot\nabla\left(\frac{1}{\bar{r}}\right)dS-2\pi\mu_P\right]+\iint_S\left(\frac{\sigma}{\bar{r}}\right)dS+\iint_W\mu_W\hat{n}\cdot\nabla\left(\frac{1}{\bar{r}}\right)dS \tag{3.25}$$

Discretizing the surface by breaking it up into panels gives the discretized form of Eq. (3.25), which allows the integrals to be evaluated as surface integrals over each panel. The surface integrals represent the velocity potential influence coefficients per unit singularity strength for panel K acting on the control point J. Hence, Eq. (3.25) becomes:

$$\sum_{K=1}^{N_S}(\mu_K C_{JK})+\sum_{K=1}^{N_S}(\sigma_K B_{JK})+\sum_{K=1}^{N_W}(\mu_{W_L}C_{JL})=0\Big|_{J=1,N_S} \tag{3.26}$$

where:

$$B_{JK}=\iint_K\left(\frac{1}{\bar{r}}\right)dS \tag{3.27}$$

and

$$\begin{gathered}C_{JK}=\iint_K\hat{n}\cdot\nabla\left(\frac{1}{\bar{r}}\right)dS\\ C_{JJ}=-2\pi\end{gathered} \tag{3.28}$$

The coefficients C_{JK} and B_{JK} represent the velocity potential influence coefficients per unit singularity strength for panel K acting on the control point panel J. Eqs. (3.27) and (3.28) are functions of geometry only and can be solved for all panels to form the influence coefficient matrix. Since the source values are known and the wake doublet values can be determined as functions of the surface doublet values, only the surface doublet strengths are unknowns. Solving for these unknown doublet strengths allows all of the panel singularity strengths to be known. From these singularity strengths, surface velocities can be determined. Using the surface velocities, the aerodynamic forces and moments can be calculated.

3.3 GRAPE

Grids will be generated for the viscous 2-D analysis using a program called GRAPE. GRAPE stands for **GR**ids about **A**irfoils using **P**oisson's **E**quation. This program was chosen for grid generation because of its ability to handle arbitrary shapes, which is important when the shape of ice on the leading edge of an airfoil is considered.

A detailed development of the theory behind this program is presented in Sorenson[35]. In this section, only the main points will be discussed.

Let $\xi = \xi(x,y)$ and $\eta = \eta(x,y)$ specify the mapping from the physical space to the computational space. The mapping functions are required to satisfy the Poisson equations:

$$\xi_{xx} + \xi_{yy} = P \tag{3.29}$$

$$\eta_{xx} + \eta_{yy} = Q \tag{3.30}$$

The following relations are useful in transforming equations between computational space and physical space:

$$\xi_x = {y_\eta}/{J} \tag{3.31a}$$

$$\xi_y = {-x_\eta}/{J} \tag{3.31b}$$

$$\eta_x = {-y_\xi}/{J} \tag{3.31c}$$

$$\eta_y = {x_\xi}/{J} \tag{3.31d}$$

where

$$J = x_\xi y_\eta - x_\eta y_\xi \tag{3.31e}$$

Applying Eqs.(3.31) to Eqs.(3.30) yields the transformed Poisson equations

$$\alpha x_{\xi\xi} - 2\beta x_{\xi\eta} + \gamma x_{\eta\eta} = -J^2(Px_\xi + Qx_\eta) \tag{3.32a}$$

$$\alpha y_{\xi\xi} - 2\beta y_{\xi\eta} + \gamma y_{\eta\eta} = -J^2(Py_\xi + Qy_\eta) \tag{3.32b}$$

where

$$\alpha = x_\eta^2 + y_\eta^2 \tag{3.32c}$$

$$\beta = x_\xi x_\eta + y_\xi y_\eta \tag{3.32d}$$

$$\gamma = x_\xi^2 + y_\xi^2 \tag{3.32e}$$

Solving Eqs. (3.32), for a particular choice of inhomogeneous terms, P and Q, and for a particular set of boundary conditions, causes a grid to be generated.

Chapter 4
Research Methodology

In this section, the proposed method will be overviewed. Then, two similar approaches will be presented and the shortcomings of these methods will be pointed out. Finally the proposed method will be developed in detail.

4.1 Overview

A quick method to analyze a three-dimensional flowfield about a complete configuration is to use a panel method, like PMARC. But PMARC has the deficiency of not being able to account for the effect of flow separation on aerodynamic characteristics. No potential flow method can effectively model the rotational flow in the separated wake region. If the wake is away from a lifting surface, the effect is not critical. However, on a lifting surface with a separated boundary layer, a better approach is necessary. Although a 3-D Navier-Stokes method with an appropriate turbulent model may be the ultimate solution, it would require a very time-consuming process to generate a proper computational grid for the flow solver of choice. The computing time for the flow field solution would be extensive. The proposed approach is to use both the panel code (PMARC) and a 2-D Navier-Stokes code (ARC2D) in the following way.

Essentially, in the near field (i.e., near the wing), the 2-D solution is valid; while in the farfield the 3-D solution is appropriate. These two solutions must be matched so the local circulation is the same. This matching condition generates effective sectional angles of attack for the 3-D boundary conditions. In this approach, the 2-D method is applied to the iced airfoil section before running the 3-D code. The 2-D results are used in an iterative manner to arrive at the converged 3-D solution. Experimental data can also be used to supply the necessary boundary *conditions to the 3-D code.*

4.2 A Similar Attempt

Because the attempt about to be reviewed uses the same tools as the proposed research, its review has been reserved until this section.

Nathman and Strash[36] proposed using ARC2D to handle the viscous effects about an iced airfoil and VSAERO to analyze the irrotational flowfield about an aircraft with iced lifting surfaces. (As previously mentioned, PMARC was developed from VSAERO.) Two methods were suggested. In the first method, ARC2D was used "to compute the separation bubble behind the artificial ice shape for various angles of attack. The geometry of this bubble was used as input to VSAERO as a type"[36] of separated wake. The second (simpler) method "used the forces predicted on the horizontal tail by ARC2D (allowing for the downwash predicted by VSAERO) and the forces of the rest of the airplane from the panel code."[36]

The problem with both of these methods is that neither method really takes the three-dimensional effects into account. In the first method, the two-dimensional solution is used to set the location for the separated wake for the three-dimensional boundary condition. In the second method, the downwash is used to correct the angle of attack for the two-dimensional solution, but the geometry of the iced shape and separated region is not allowed to affect the three-dimensional solution. The proposed method will allow the two-dimensional flowfield to affect the three-dimensional flowfield and vice versa.

4.3 Coupling a Panel Method with 2-D Navier-Stokes Solutions

Two-dimensional Navier-Stokes solutions have been shown to provide reasonably good prediction of icing effects on airfoil sections[15,27].

Theoretically, the 2-D solution is valid in the near field (i.e., near the wing section); but in the far field, the 3-D method (PMARC) must be used. These two solutions are to be matched by requiring the local circulation to be the same. This matching condition generates effective sectional angles of attack for the 3-D boundary conditions. Note that this approach is not the same as the classical nonlinear lifting-line theory because no lifting-line method is used. This method has been used successfully in predicting the aerodynamics of a variety of airplane configurations at different angles of attack, even beyond stall[37,39,40]. Note that if the 2-D results are directly used, the method becomes quasi-two-dimensional. The three-dimensional effect cannot be properly accounted for.(E.g, Ref. 36) In the proposed approach, the 2-D method (ARC2D) is applied to an airfoil section independently of the operation of the 3-D code(PMARC). Then the results from the 2-D analysis will be supplied to PMARC in the form of a lookup table. The lifting surfaces can be discretized in to chordwise strips (like airfoil sections) that have an effective angle of attack. At a given effective angle of attack, the 2-D values of lift, drag and pitching moment can be looked up and applied.

In this method (The following description is adapted from Tseng and Lan[39].), an effective angle of attack at a spanwise station, α_e , is calculated based on the geometric angle of attack, α_n, the induced angle of attack, α_i, the zero lift angle of attack, α_o, and viscous effects, $\Delta\alpha$.

Therefore:

$$\alpha_e = \alpha_n - \alpha_i - \alpha_o - \Delta\alpha \tag{4.1}$$

From this it follows that

$$c_{\ell(3-D)} = c_{\ell_\alpha} \sin(\alpha_n - \alpha_i - \alpha_o - \Delta\alpha) \tag{4.2}$$

Assuming $c_{\ell_\alpha} = {}^{2\pi}\!\big/\!\sqrt{1-M_\infty^2}$, Equation (4.2) can be solved for α_i:

$$\alpha_i = \alpha_n - \sin^{-1}\left[\frac{c_{\ell(3-D)}}{c_{\ell_\alpha}}\right] - \alpha_o - \Delta\alpha \tag{4.3}$$

If we let the 2-D section lift coefficient, evaluated at α_n - α_i, equal $c_{\ell(2-D)}$, then define:

$$f = \frac{c_{\ell(2-D)}}{c_{\ell(3-D)}} \tag{4.4}$$

Since $c_{\ell(3-D)}$ is computed with an inviscid theory, its value is usually larger than $c_{\ell(2-D)}$ if $\Delta\alpha = 0$. Therefore, f is usually less than 1.0. In this case, a geometric angle of attack (α') which produces the reduced lift can be found. That is: $\sin\alpha' = f \cdot \sin\alpha_n$ or,

$$\alpha' = \sin^{-1}(f \sin\alpha_n) \tag{4.5}$$

It follows that $\Delta\alpha$ in Equation (1) becomes

$$\Delta\alpha = \alpha_n - \alpha' \tag{4.6}$$

The solution is obtained iteratively as follows:

1. Assume $\Delta\alpha = 0$.
2. Find α_i from Equation (4.3).
3. Calculate f from Equation (4.4).
4. Determine $\Delta\alpha$ from Equations (4.5) and (4.6).

5. Use $\Delta\alpha$ to reduce α in the 3-D boundary condition to determine $c_{\ell(3-D)}$.

6. Repeat steps 2 through 5 until the successive total lift coefficients differ by less than a small value, e.g. 0.5%.

Figure 4 is a graphical representation of this iterative process.

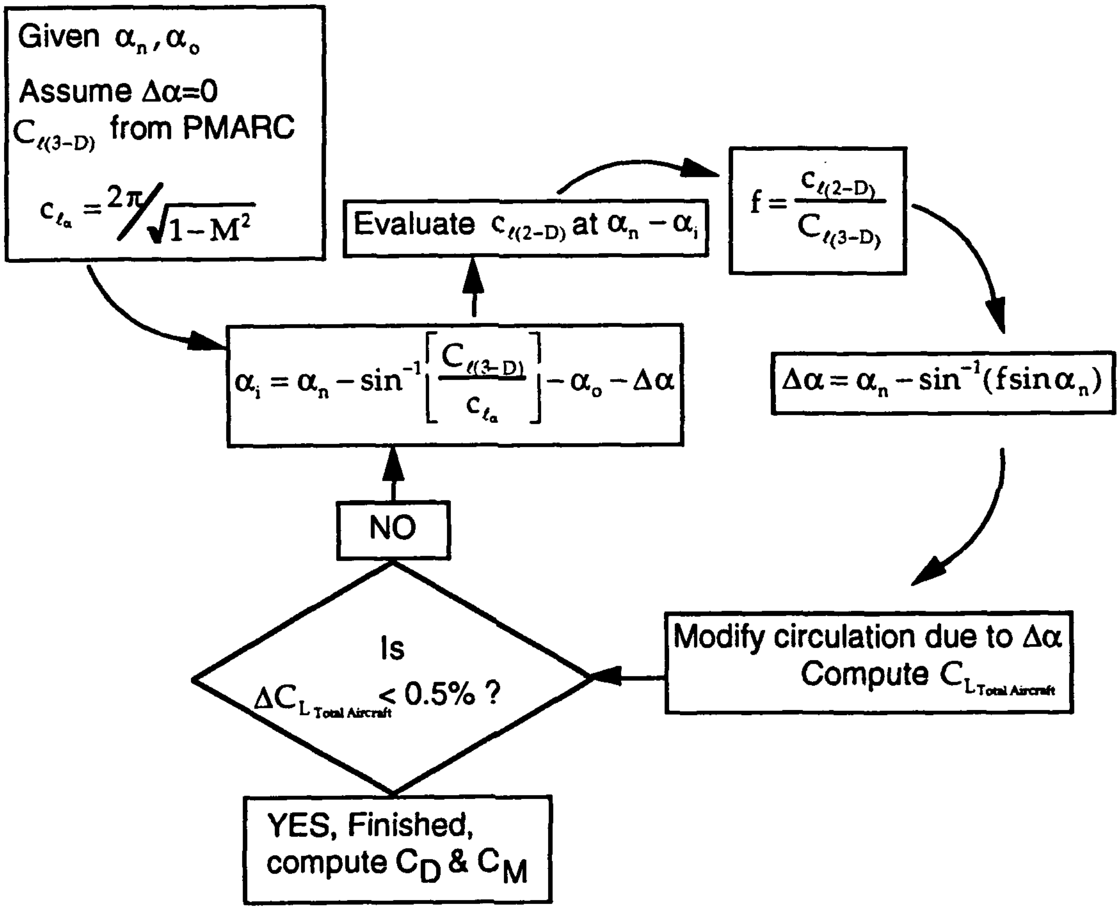

Figure 4. Schematic of Solution Process to Include Vicous Effects

Following the satisfaction of the convergence criterion of Step 6, drag and pitching moment are calculated. The associated drag and pitching moment for each $c_{\ell(2-D)}$ used in (4.4) is combined with the forces from the converged inviscid solution to provide C_D and C_M.

C_D is a combination of the inviscid induced drag calculated by PMARC and the viscous sectional drag (included only for lifting surfaces). The viscous sectional drag is evaluated, like $c_{\ell(2-D)}$ at $\alpha_n - \alpha_i$. The calculation of C_D is illustrated in Eq. (4.7).

$$C_D = C_{D_{non-lifting}} + \left(\sum_{i=1}^{n} c_{d(3-d)_i} + c_{d(2-d)_i} \cdot \left(\frac{S_i}{S_{ref}} \right) \right), \tag{4.7}$$

where n is the number of spanwise stations on the lifting surfaces, S_i is the area of spanwise stations for a lifting surface. Remember, viscous drag for a non-lifting surface, e.g. the fuselage, is not accounted for in the present investigation.

C_M is a combination of the moment due to inviscid lift calculated by PMARC and the viscous drag plus the zero-lift pitching moment of the spanwise station being considered. Equation (4.8) illustrates the calculation of C_M.

$$C_M = C_{M_{non-lifting}} + \sum_{i=1}^{n} \left(c_{m_o(2-d)_i} + \frac{\left(c_{d(2-d)_i} \cdot z_{cg_i} - c_{l(3-d)_i} \cdot x_{cg_i} \right)}{\bar{c}} \right) \cdot \left(\frac{S_i}{S_{ref}} \right), \tag{4.8}$$

where $C_{M_{non-lifting}}$ is the pitching moment coefficient contribution calculated by PMARC for non-lifting surfaces, n is the number of spanwise stations on the lifting surfaces, and x_{cg_i} and z_{cg_i} are the perpendicular x and z distances from the moment reference for the 2-d data to the moment reference of the model.

To show the applicability of the forces and moments obtained from the 3-D code, these forces and moments (in non-dimensional form) could be used as input for a simulator analysis. This analysis would show how the aircraft would react in a specified flight condition with a representative ice accumulation. This leads directly to the partial satisfaction of the certifying regulations.

Chapter 5
Results and Discussion

This chapter presents the results and discussion of the present study. This chapter is broken into two parts: the first will discuss the viscous analysis of the flow about an iced airfoil, using ARC2D; the second will discuss the analysis of four configurations using PMARC with viscous corrections: wing alone, wing-body, wing-fuselage-low horizontal tail and wing-fuselage-high horizontal tail.

5.1 Airfoil Viscous Analysis

This section presents work and results completed to learn the workings of the viscous code, the grid generation code, then the viscous analysis of an airfoil section with an iced leading edge.

5.1.1 NACA 0012 Study

Initially, to become familiar with both the grid generation program, GRAPE, and the Navier-Stokes code, ARC2D, a study was conducted on the NACA 0012. This airfoil was chosen for two reasons:

(1). experimental data is readily available (e.g., Abbott and von Doenhoff[38], e.g.);

(2). This is the base airfoil on which experimental as well as analytical icing research has been conducted.(see Bragg[20], Korkan, et al.[23], Bragg, et al.[25] & Potapczuk[27]) Thus to put an iced shape on this airfoil will be a natural development of previous research.

In this study, parameters, such as Mach number, R_e etc., were chosen to match the data of Ref. 38. Figure 5 shows the C-grid system used to characterize the flow field around the NACA 0012. (Every fourth grid point is shown to improve clarity.) The grid is made up of 253 points in the direction around the surface of the airfoil and 64 normal to the surface. There are 46 points in the wake, leaving 207 on the surface. Grid points are concentrated near the leading and trailing edges of the airfoil.

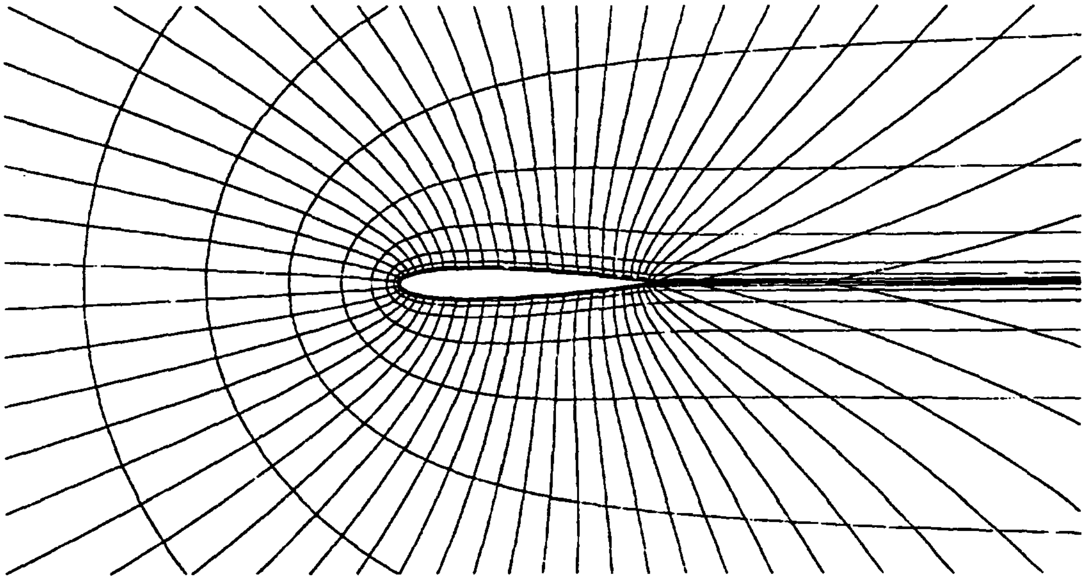

Figure 5. Grid About a NACA 0012

While refining this grid, sensitivity studies were performed to determine the area required to return to freestream conditions and the grid density in the spacing of the first grid point normal to the surface. In the grid area sensitivity study, flow fields of 10c X 10c and 15c X 15c were compared. Comparing the values at $\alpha=2°$ (presented in Table 1), one can see that both flow fields model the physical situation with the same accuracy. Thus, the flow field with the smaller area will be used throughout this study.

Table 1. Comparison of Grid Area Sensitivity, α=2°

	c_l	c_d	c_m
10c X 10c	0.18945	0.01667	0.00876
15c X 15c	0.18965	0.01615	0.00741

When comparing the spacing of the first grid point, the parameter used to control this is Δs. This value is given in percent chord and is used to set the spacing of the rest of the points in the direction normal to the surface out to freestream. In this study, two values of Δs were chosen. Both values put a sufficient number of points in the boundary layer to characterize the viscous nature of this area of the flow field. The tradeoff here is between the number of points to characterize the boundary layer vs. the number of points to characterize the flow outside of the boundary layer. Table 2 shows a sample of the results of this study at $\alpha = 2°$.

Table 2. Comparison of First Grid Point Spacing, α=2°

	c_l	c_d	$c_{m_{c/4}}$
Abbott and von Doenhoff[38]	0.21	0.010	0.00
Δs=0.0005	0.20931	0.00692	0.00363
Δs=0.00005	0.18945	0.01667	0.00876

Though spacings for both grids do a satisfactory job of predicting c_l, the densest spacing overpredicts both c_d and $c_{m_{c/4}}$. Thus the less dense spacing will be used throughout this study.

A third sensitivity study involving the number of iterations to convergence was conducted. In this study a variable timestep was used based on local eigenvalues. Two timesteps were considered: Δt=2.0 and

Δt=4.0. **The convergence rates to a steady state value are compared for c_l, c_d, and $c_{m_{c/4}}$ in Figure 6.**

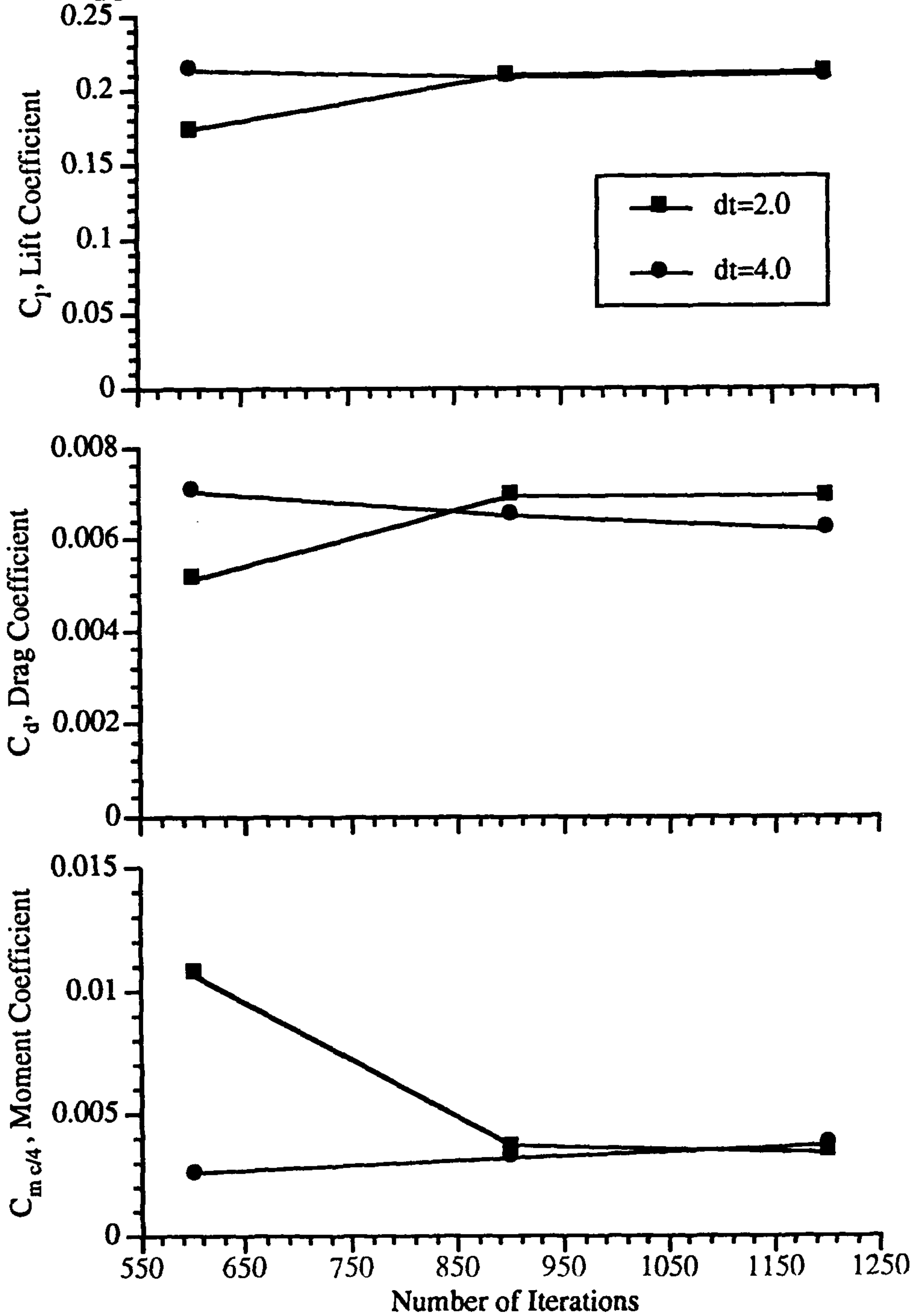

Figure 6. Convergence Study for Variations in Timestep

Lift coefficient has converged for Δt=4.0 in 900 iterations (and possibly as early as 600 iterations). Drag coefficient makes a small change (<3%) after 900 iterations at Δt=4.0. The moment coefficient shows convergence in 900 iterations for this longer timestep also. A longer timestep was not considered and the shorter timestep required more CPU time to reach convergence. Thus, this longer timestep will be used for computations, unless convergence plots show the need for further calculation.

An overall comparison of the ARC2D calculations of the lift, drag and pitching moment coefficients with experimental data are given in Figure 7a-c. This comparison is made using the "standard roughness" values of Abbott and von Doenhoff[38] and with ARC2D, assuming transition at the point where the roughness was applied in the experiment. Reynolds numbers were matched for this study at $R_e = 6 \times 10^6$.

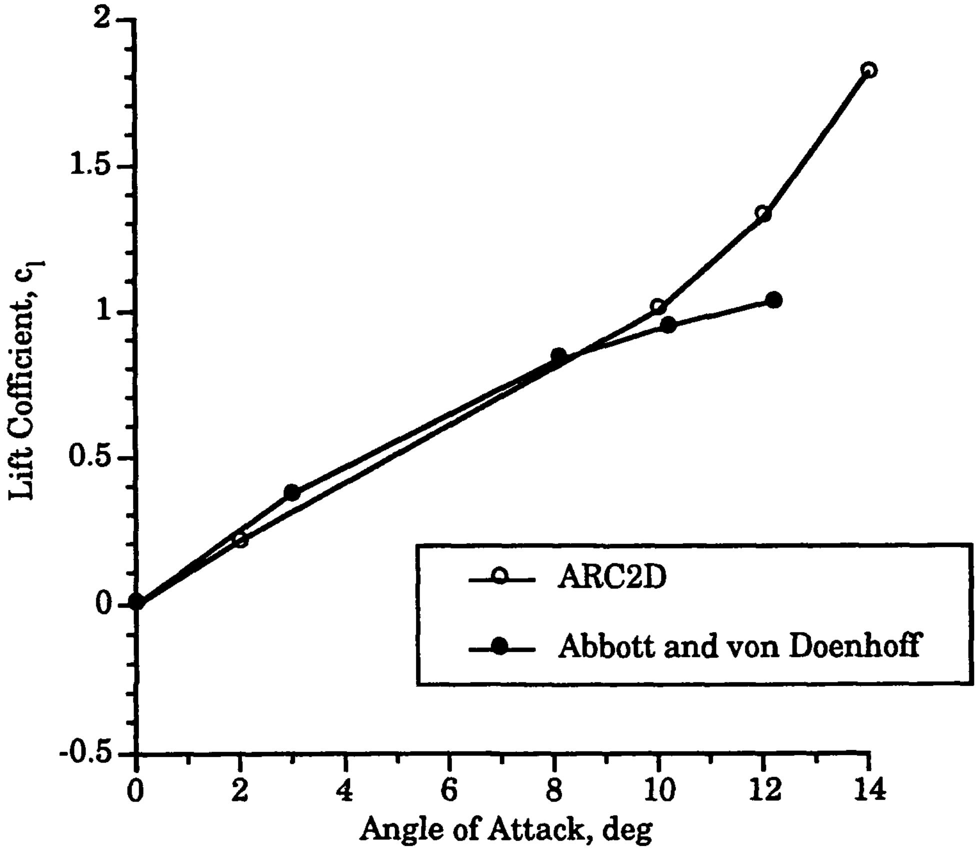

Figure 7a. Lift Coefficient Comparison of ARC2D with Reference 38

NACA 0012 Section, $R_e = 6 \times 10^6$

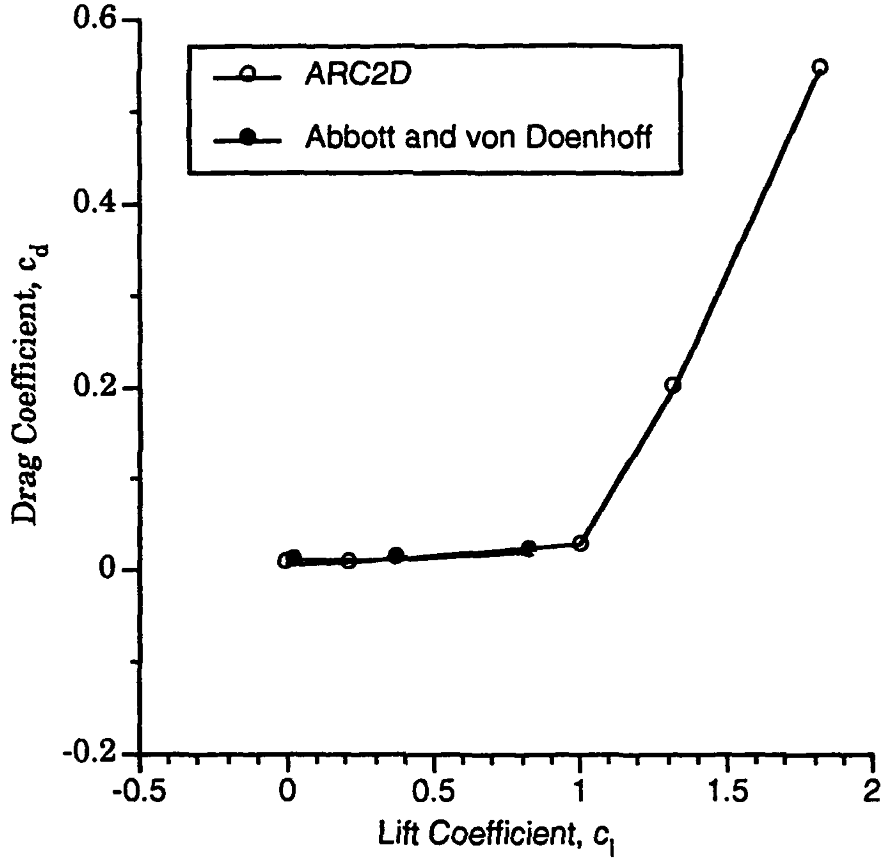

Figure 7b. Drag Polar Comparison of ARC2D with Data of Reference 38

NACA 0012 Section, $R_e = 6 \times 10^6$

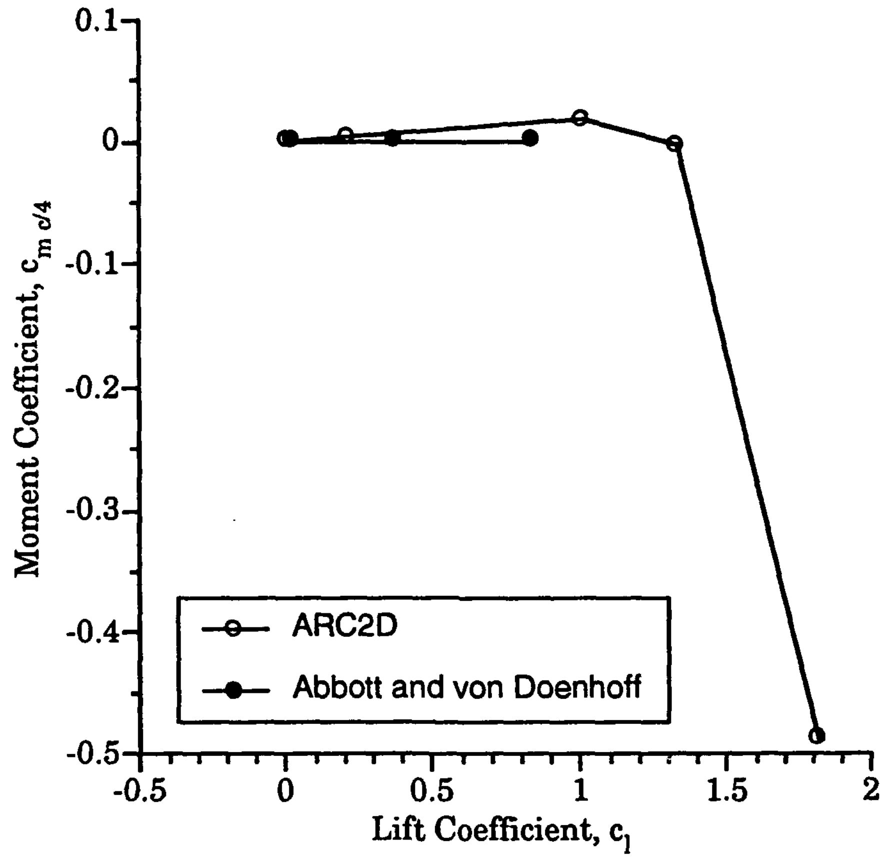

Figure 7c. Moment Coefficient Comparison of ARC2D with Data of Reference 38

NACA 0012 Section, $R_e = 6 \times 10^6$

Figure 7 shows that ARC2D data compares well with experimental data up to stall. Note: In Abbott and von Doenhoff[38], for "standard roughness", drag and pitching moment coefficients are not presented beyond a lift coefficient of approximately 0.8.

5.1.2 Iced NACA 0012

Following this study, a generic ice shape was attached to the leading edge of the NACA 0012. This ice shape was developed by Bragg, et al.[19] and used in the studies by Potapczuk[27]. Freestream conditions for the development of this ice shape were $\alpha=4°$, T=18°F, V=130 mph. Ice accretion time was 5 minutes. To accurately model this shape, a picture of this shape from Potapczuk[27] was electronically scanned, then prominent points were digitized producing the ice shape attached to the NACA 0012. The combined "airfoil" (airfoil plus ice shape) is presented in Figure 8. Note the similarity of the ice shape with the shapes in Figure 1.

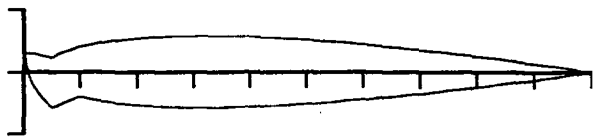

Figure 8. Iced Shape on a NACA 0012

A grid was generated about this iced airfoil using the results of the previous sensitivity studies and the same grid point arrangement of the previous NACA 0012 study. (This is the same arrangement as Potapczuk[27].) This grid is shown in Figure 9. (Every fourth grid point is shown to improve clarity.)

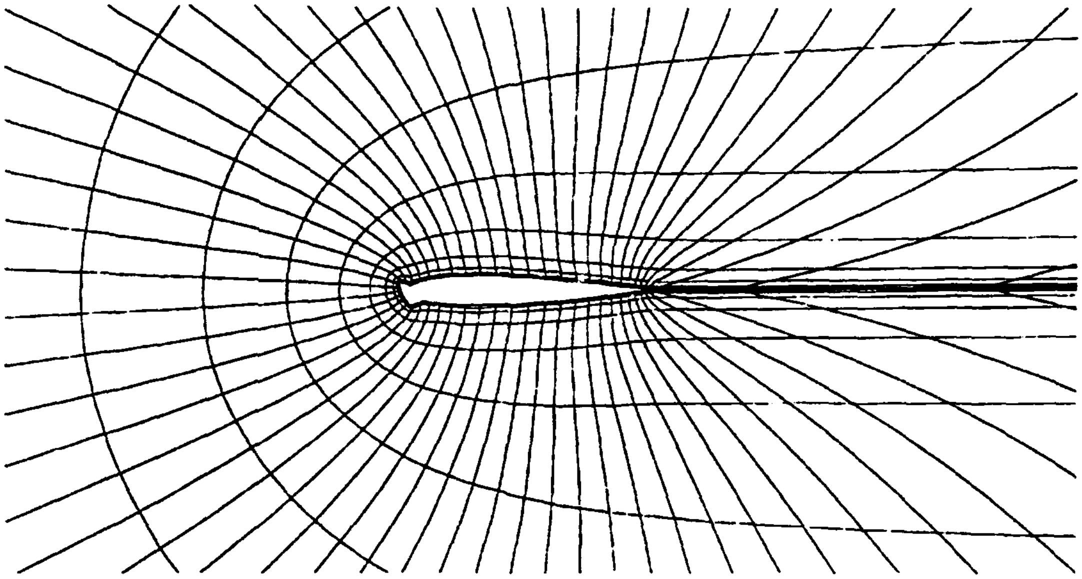

Figure 9. Grid About a NACA 0012 with an Iced Shape on the Leading Edge

After matching the flow conditions of Bragg[20] and Potapczuk[27], an angle of attack sweep was conducted. The results are compared with Bragg[20] and Potapczuk[27] in Figure 10.

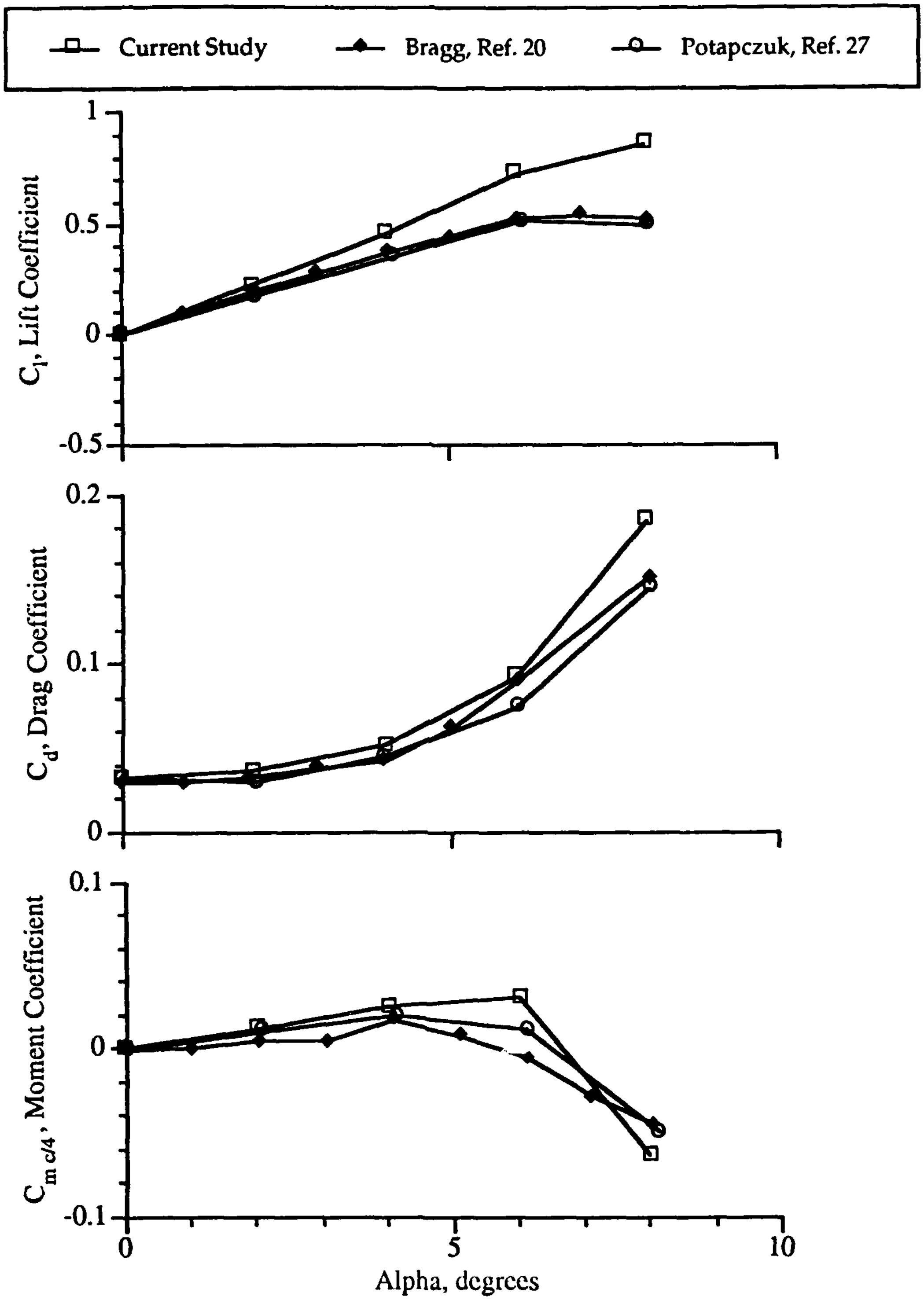

Figure 10. Comparison of Present Study with Bragg[20] and Potapczuk[27] Iced NACA 0012 Section, $R_e = 1.5 \times 10^6$

The modeling is good up to the onset of stall. At that point, both the data of this research and Potapczuk[27] do not model the flow field well. To achieve the lift coefficient at α=8°, Potapczuk[27] averaged "the pressure coefficient at each location over several shedding periods."[27] This is similar to an experimental setup where only the steady pressures are measured. (Due to the response times of the measuring equipment, the experimental setup can not measure the unsteady pressures.) Potapczuk[27] attributes this C_l discrepancy to the turbulence model and suggests trying a different turbulence model to capture the physics of the flow field. Potapczuk[17] states that the k-ε and Johnson-King turbulence models have been shown to have little effect.

5.1.3 Turbulence Modeling

The Baldwin-Lomax turbulence model is a two-layer algebraic turbulence model, where $\mu = \mu_l + \mu_t$. The turbulent viscosity coefficient, μ_t, is computed using two different formulae for inner and outer layers of the boundary layer. The switch is made at the height above the surface where the coefficients from the two regions match. The problem arises in the computation of μ_t for the outer layer. The vorticity for the outer layer is based on choosing a length scale, y_{max}, when the moment of vorticity, $F(y)$, is a maximum, F_{max}. A more detailed discussion of the formulation is given in References 32 and 41.

One problem with the current formulation of this turbulence model is that two (or more) extremes for F can occur and the current search routine picks the greatest. Figure 11 is a plot of vorticity magnitude(ω) and moment of vorticity, F(y), vs. y/c, a distance measured normal to the airfoil surface (in this case at the point of maximum thickness for the NACA 0012). (F_{max} is depicted by the location of y_{max}. See Figure 11.) The value picked by the current search scheme is not always representative of the physical situation. It might pick an F that is in the boundary sublayer or one that is outside of the boundary layer (which it appears to have picked here). In Figure 11, F_{max} chosen by the current search routine is outside the boundary layer. The length scale for the moment of vorticity is too great. This does not properly model the physical situation by causing "the details of the computed flow to be distorted or washed out".[41] Thus, as part of the present study, the current implementation of the Baldwin-Lomax model has been found to not model the physical situation well near stall. A modification to the search routine could improve the results of Figure 10 near the stall.

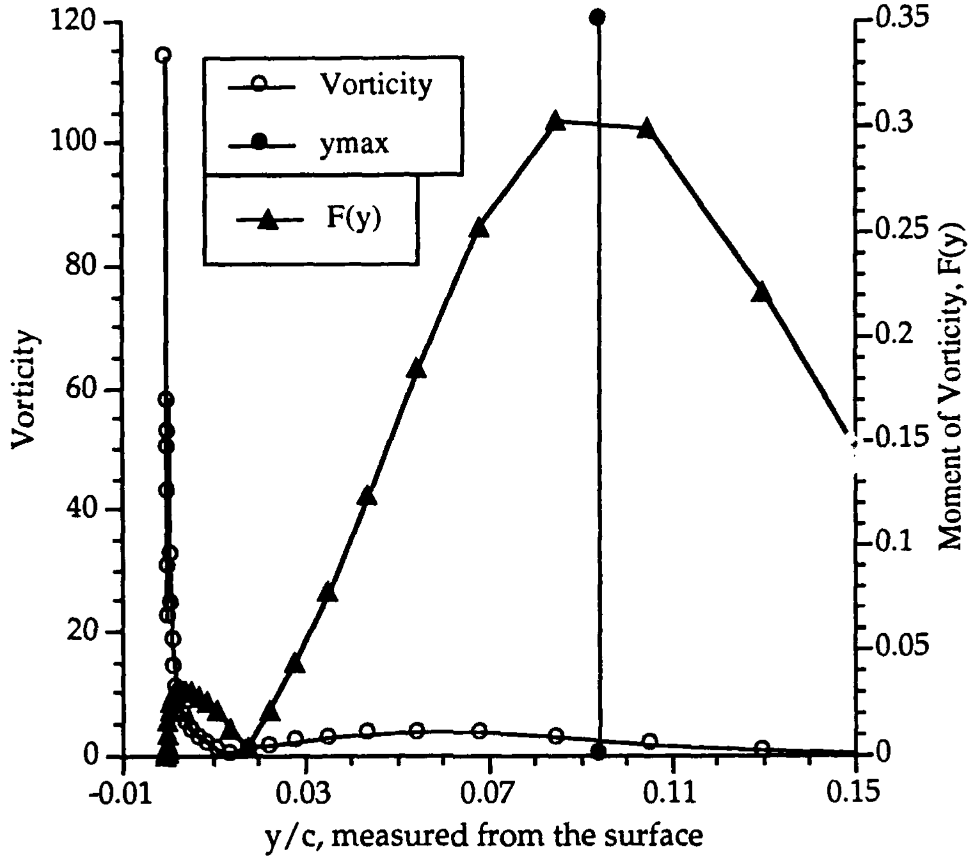

Figure 11. Vorticity and Moment of Vorticity vs. y/c Measured From the Surface of an Iced NACA 0012 at t/c_{max}

This study has applied the "modified" Baldwin-Lomax turbulence model[41] in ARC2D and found some improvement in the modeling of the flow situation. In the modified Baldwin-Lomax model, numerous relative maxima are allowed and the first maximum outside of the sublayer is chosen. In Figure 12, the second of 6 maxima is chosen to compute the eddy viscosity. (F_{max} #3 through #6 are at a location, y/c, greater the scale of this figure.) This data was taken near t/c_{max} on the upper surface of the iced NACA 0012, the region where the trapped vortex has been observed. Note the location is in the boundary layer and outside of the sublayer. The computed lift coefficient at $\alpha=8°$ is closer to the experimental value. But (using PLOT3D, "a computer graphics program designed to visualize the grids and solutions of Computational Fluid Dynamics"[42]) there still appears to be a trapped vortex at approximately t/c_{max}.

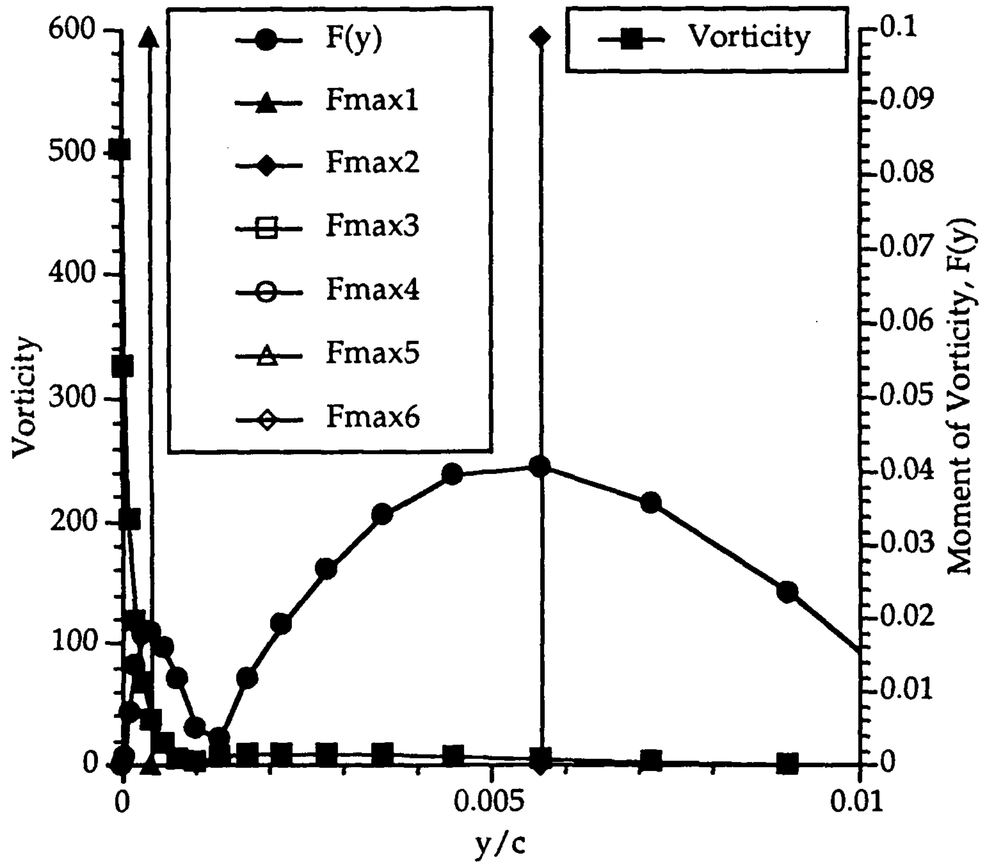

Figure 12. Choice of F_{max} in the Application of the Modified Baldwin-Lomax Turbulence Model

Now the question can be asked: Is this vortex trapped or is the numerical method looking at only one instant in time and only seeing the vortex where it sits?

To answer this question a study was conducted to make time accurate calculations using ARC2D. (A Time Accurate Solution is one of the solution options in ARC2D.) Zaman and Potapczuk[43] used this method and found a pє riodic nature to the flowfield. (See Figure 13.)

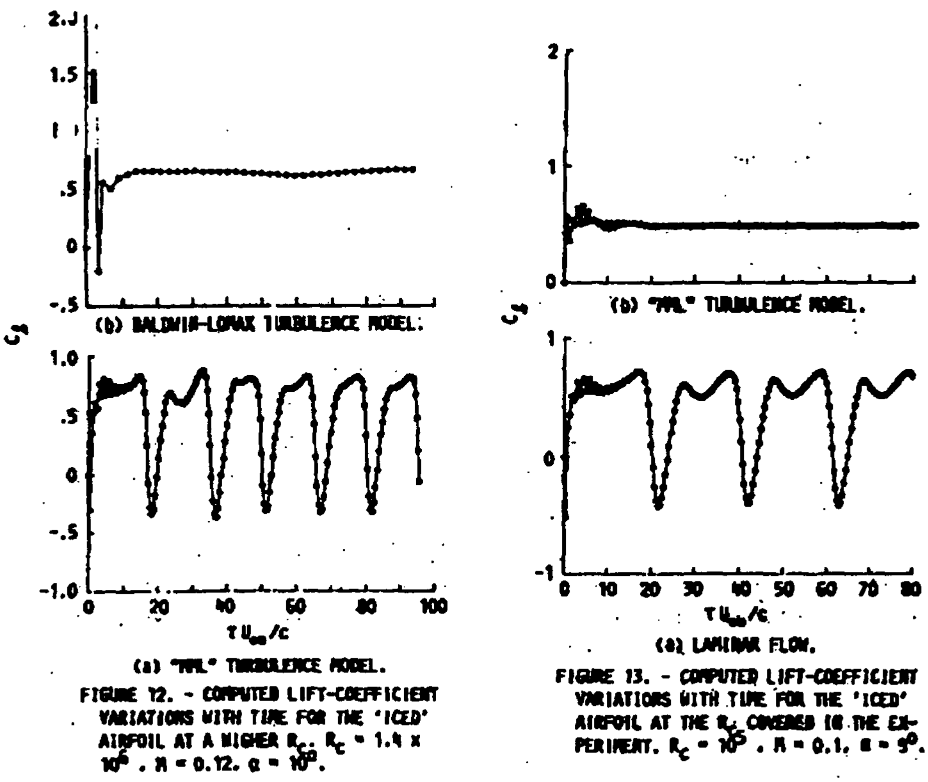

Figure 13. Computed Lift Coefficient Variations with Time

(Copied from Reference 47)

In this study, Zaman and Potapczuk found a vortex being shed off the upper horn and convected downstream along the upper surface. This produced a periodic loading showing up in variations in the lift and pitching moment coefficient. If one could take this periodic loading and use a time averaging scheme, the experimental data collection could be simulated. (E.g., See Bragg[20].)

Along this vein, the current study tried to duplicate Zaman and Potapczuk's results--hoping to generalize it for any airfoil near stall. The grid for this study over the iced NACA 0012 airfoil is the same as has been discussed previously. In the time accurate study, two modified versions of the Baldwin-Lomax turbulence model were employed. In the first, if more than one peak was present for F(y), the second peak was used (analogous to the method of Reference 41). As Degani and Schiff[41] have shown, the use of the Baldwin-Lomax turbulence model with the second peak chosen seems to more accurately model the physical situation and is the first model used in the time accurate study. In the second version, the first peak was used exclusively. Two time steps were also considered in this study: $\Delta t=0.001$ and $\Delta t=0.01$. The first was chosen to match the time step of Potapczuk[44]. The second was chosen because of the advantage of one timestep using the second Δt equals ten timesteps with the first. (This saves computing time.) Therefore, if similar lift values are found for the same time interval, the second time step could be used to move the solution away from the point when the airfoil is introduced in the flowfield. The finer timestep could be used to resolve the unsteadiness in the flowfield by using the restart option with this timestep.

Figure 14 is representative of computations of this study into the unsteady nature of the lift coefficient using a timestep of $\Delta t = 0.01$.

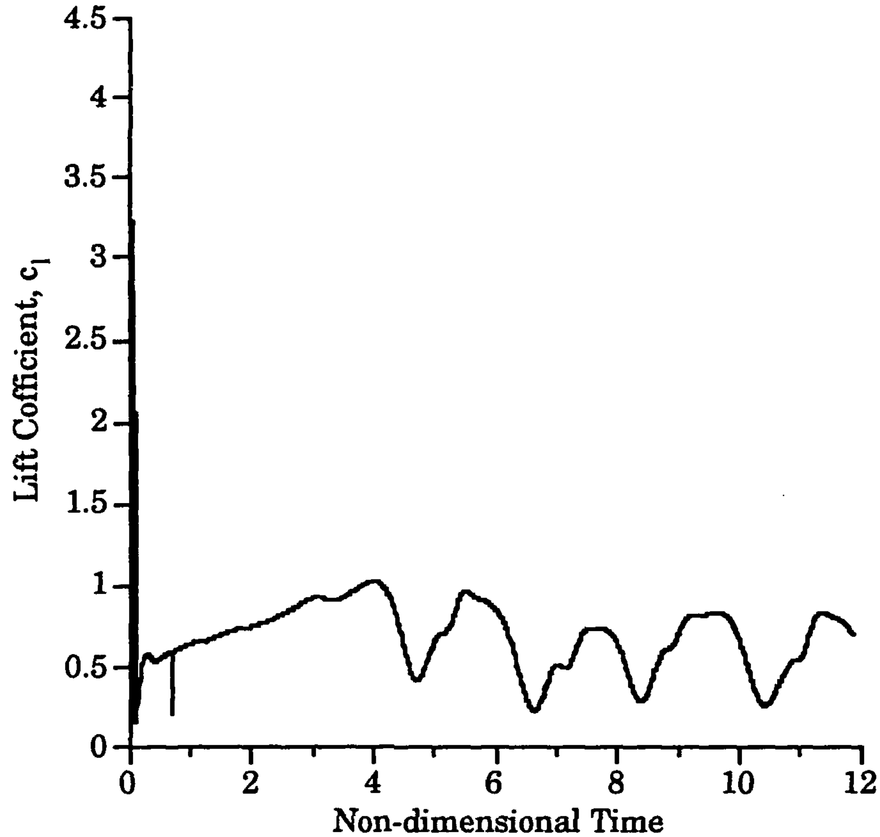

Figure 14. Example of Time Accurate Computations for Lift Coefficient, $\Delta t = 0.01$, $\alpha = 8°$

For each timestep (iteration), the airfoil, with a coarse grid, was introduced into the flowfield, then after 100 timesteps the airfoil, with a fine grid, was introduced. As can be observed over the time interval displayed, an unsteady character to the lift coefficient is evident, but a periodic character is not. Notice that it took about 5 time units for any unsteady phenomena to develop in the time histories. After the unsteady phenomena developed, the data did not demonstrate any periodicity as Zaman and Potapczuk had found. (In Zaman and Potapczuk's paper, there were times when they found this periodicity and other times when they couldn't, depending on "the turbulence model in use as well as the Reynolds number"[43].) Because of the excessive computer time involved (The timescale in Figure 14 represent 10,100 iterations. 10,100 iterations, no matter what the timestep, take about 14 hrs. on the IBM 320 RISC/6000 workstation. If ran on the VAX 9000, the University of Kansas' mainframe, these same calculations would take about 7 hrs. if one could get 100% of the CPU time.) and the lack of limited indications of reaching an unsteady solution that was pertinent to this study, the timestep with $\Delta t = 0.001$ was soon dropped. All further unsteady investigations were made with $\Delta t = 0.01$.

Using the previously determined timestep, a time averaging scheme was employed. In the time averaging scheme, the c_l for each time step is added to all of the previous and averages over the number of time steps. This scheme gives a "running average" of the coefficients. Figure 15 is an example of data from this study for $\alpha = 8°$.

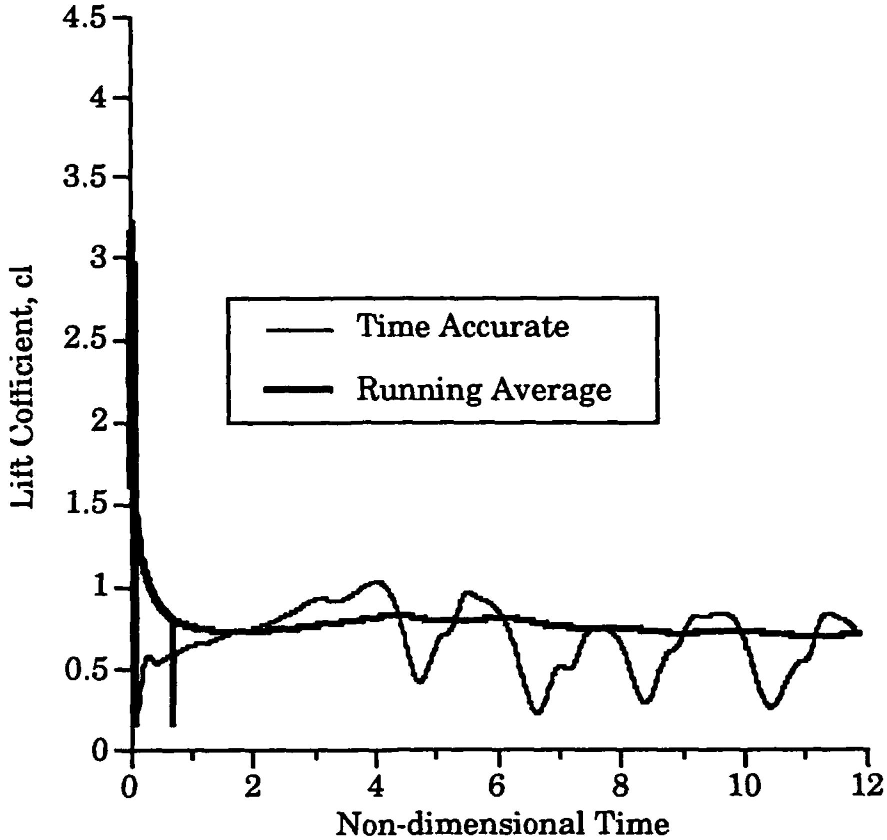

Figure 15. Lift Coefficient For An Iced NACA 0012 at $\alpha = 8°$, Time Accurate and Running Average Shown

Figure 15 shows both the time accurate results and the running average results. Note how the running average damps out the unsteady nature of the time accurate computations, acting just like a pressure sensing port on an airfoil model in the wind tunnel. The angle of attack presented is near stall and the unsteady nature of the flowfield is expected from the previous discussion, but what happens at a lower angle of attack, say $\alpha = 4°$?

Figure 16 is also a time accurate plot of lift coefficient, except $\alpha = 4°$. Note that there still is an unsteady nature to the time accurate computations, but the running average c_l matches the c_l which was computed using steady state methods in Figure 9. Thus, this "running average" scheme seems to affect the lift coefficient in the regime where the flow is truly unsteady, but has little effect on the lift coefficient where there is not massive separation.

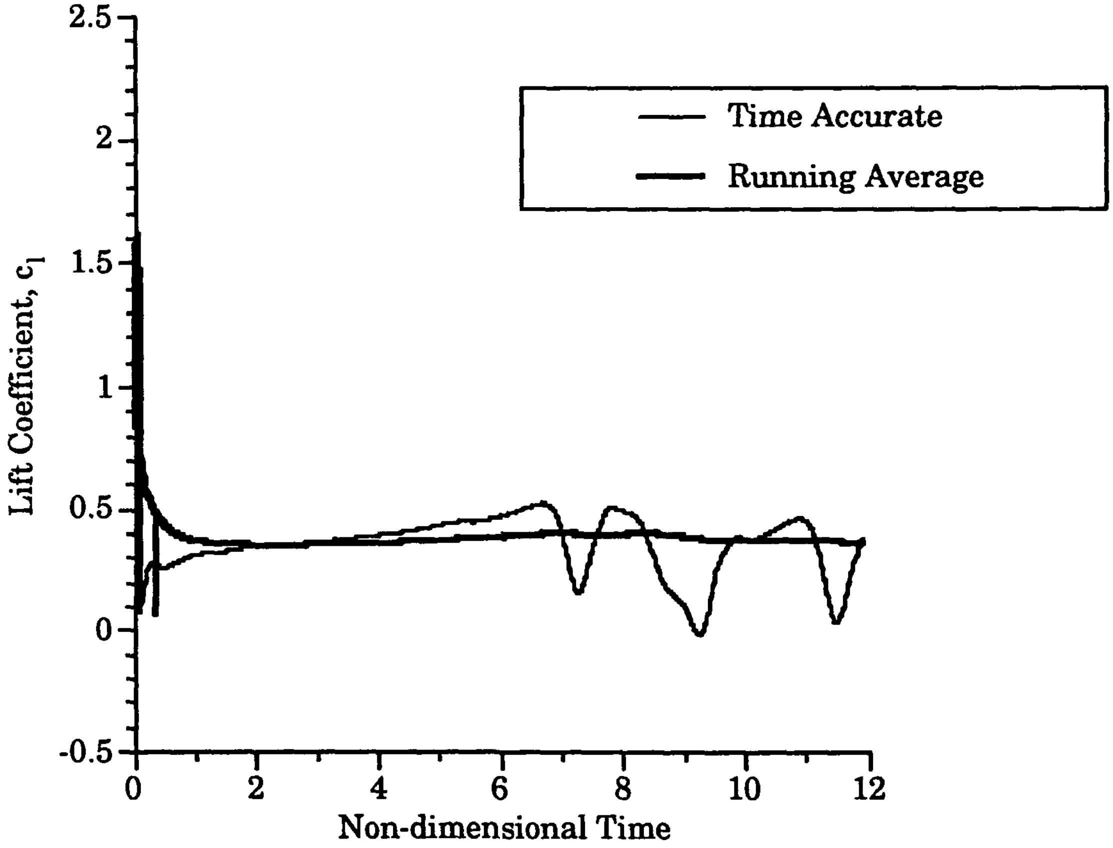

Figure 16. Lift Coefficient For An Iced NACA 0012 at $\alpha = 4°$, Time Accurate and Running Average Shown

If one is only after the global results, this technique shows promise. Unfortunately, this technique did not work as well at $\alpha = 8°$, as Figures 17a-c demonstrate. These figures are the best estimate from this study for the lift and moment curves and drag polar for the iced NACA 0012. Thus further research is necessary in this area of viscous flow calculations and will be addressed in the recommendations.

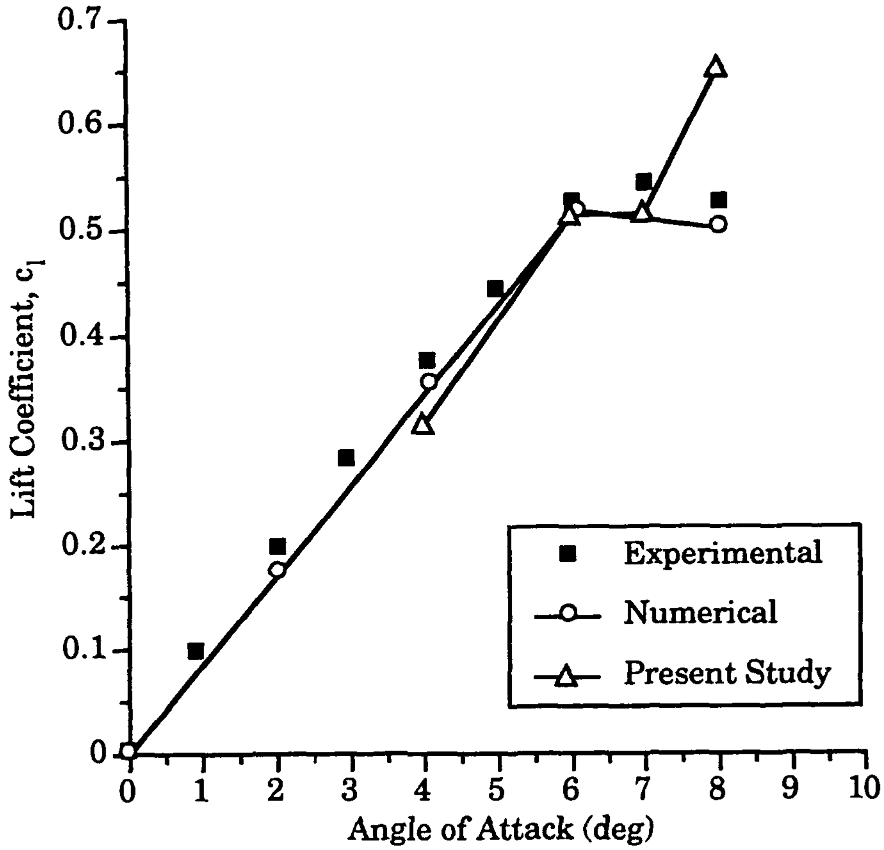

Figure 17a. Lift Coefficient for an Iced NACA 0012 Airfoil Using Time Averaging of Time Accurate Computations

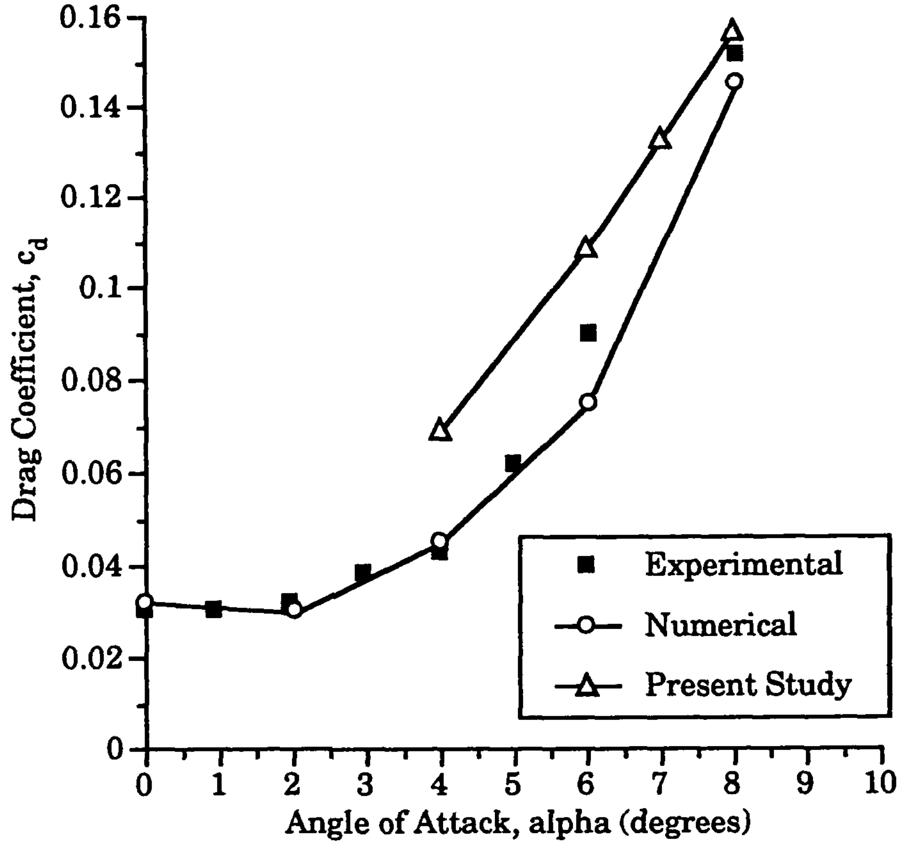

Figure 17b. Drag Polar for an Iced NACA 0012 Airfoil Using Time Averaging of Time Accurate Computations

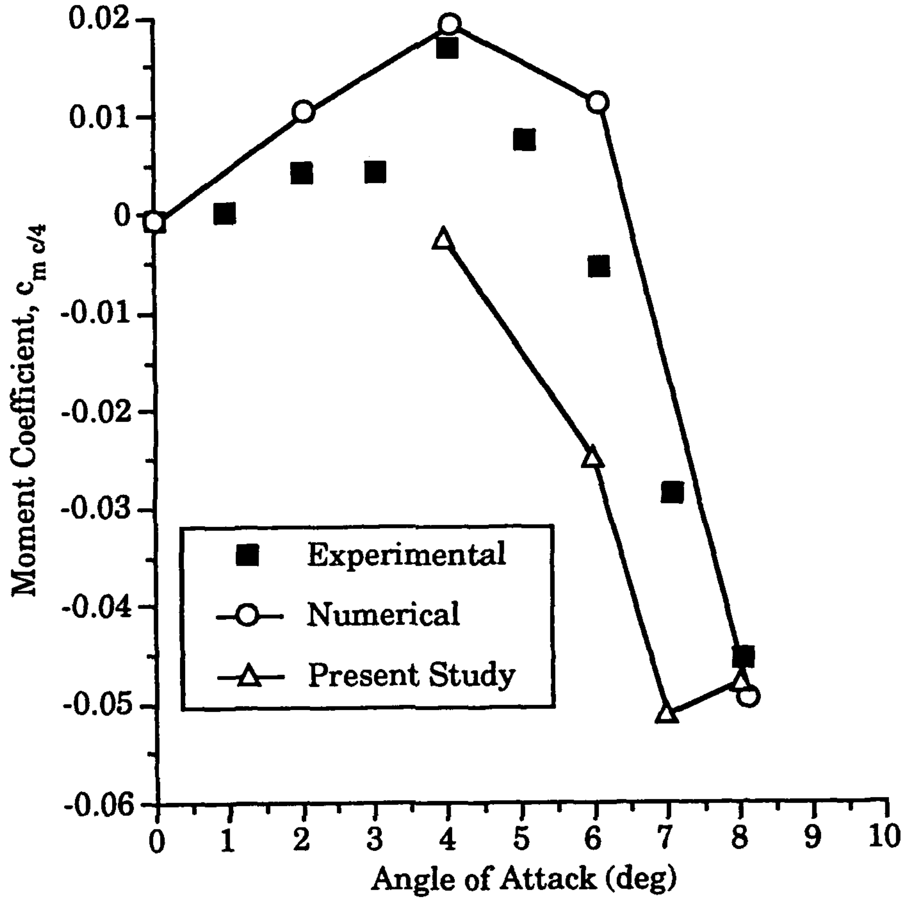

Figure 17c. Moment Coefficient for an Iced NACA 0012 Airfoil Using Time Averaging of Time Accurate Computations

5.2 Modified PMARC Results

This section deals with the results for the modifications to PMARC which include viscous effects. This section will be broken into three parts: wing alone results, wing-body results and wing-body-tail results.

5.2.1 Wing Alone

As outlined in Section 4.3, PMARC was modified to adjust the spanloading for the effects of viscosity. The first test of this method was a rectangular wing with a NACA 0012 airfoil section and an $AR = 5$. Figure 18 shows the paneling arrangement for this wing.

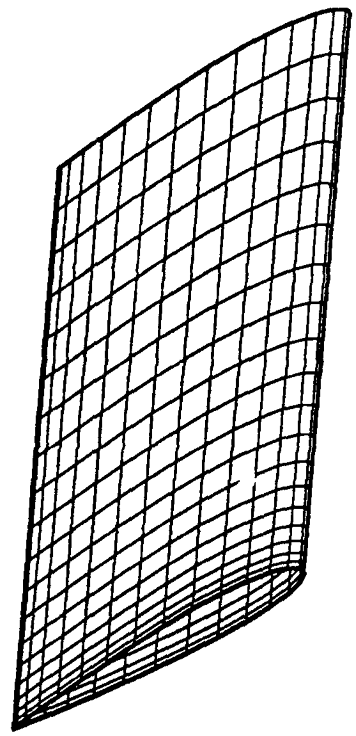

Figure 18. Panel Arrangement for A Rectangular Wing, AR = 5, NACA 0012 Section

In laying out the panels for this study, two sensitivity analyses were conducted. In the first, lift and moment coefficient sensitivity to spanwise panel number variation was investigated. 30 panels were used in the chordwise direction. Figure 19 shows the results of this study. With the number of chordwise panels chosen there seems to be a slight decrease in lift coefficient and a slight increase in moment coefficient until approximately 20 panels were used in the spanwise direction.

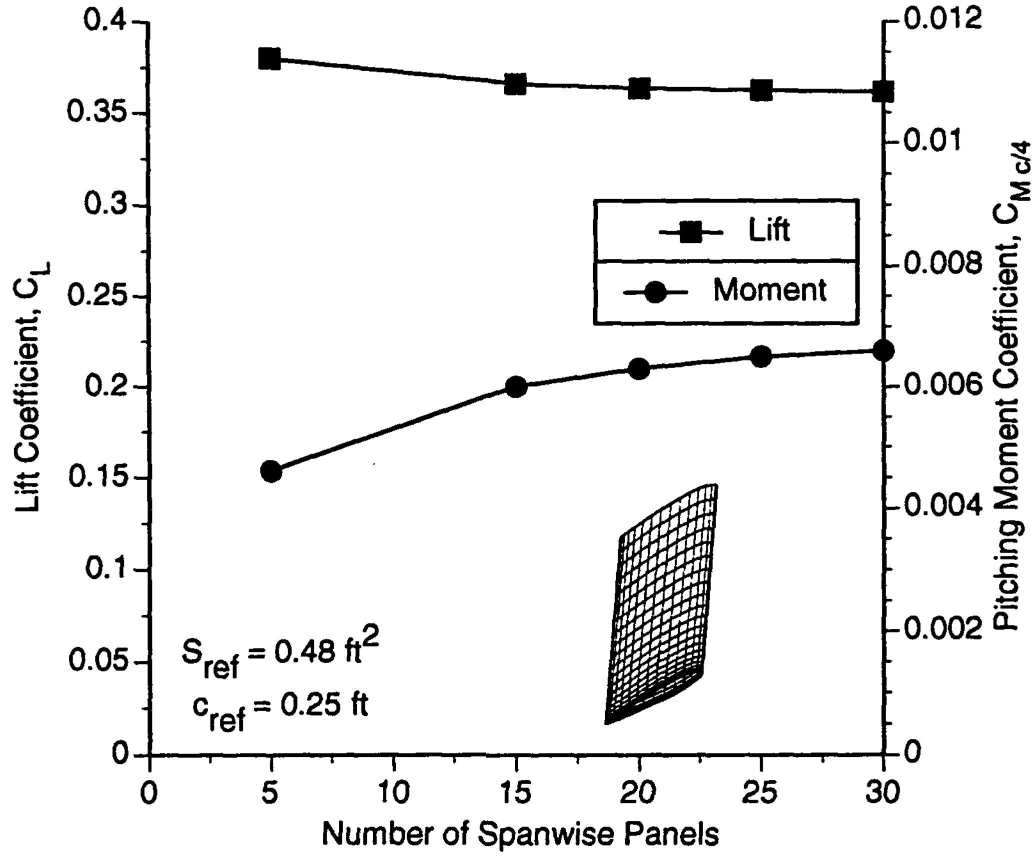

Figure 19. Sensitivity Study for Spanwise Panel Variations
α=5°, AR=5, 30 Panels Used Chordwise

The next sensitivity study involved the variation in the number of panels chordwise. The spanwise distribution was set at 20 panels with a "half-cosine" distribution. This distribution involves more coarse spacing at the wingroot and denser spacing at the wingtip using a cosine distribution. Figure 20 shows the results of this study. The x-axis displays the total number of panels in the chordwise direction. The panels were spaced using a "full cosine" spacing where paneling is more dense near the leading and trailing edges and coarser at midchord. Notice that the curve for lift coefficient "levels off" between 30 and 40 panels. The pitching moment does not appear to reach a constant value until about 80 panels. In an attempt to maintain a reasonable amount of computing time, the lower value was used for further computations.

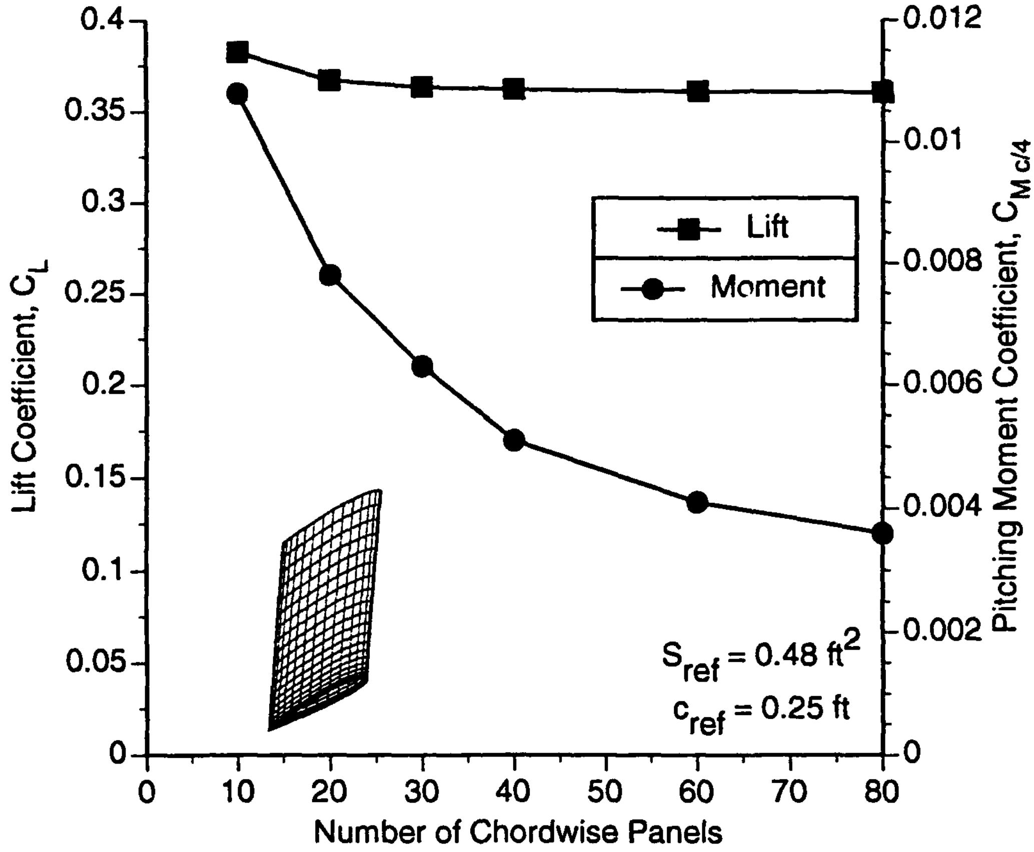

Figure 20. Sensitivity Study for Chordwise Panel Variations
$\alpha = 5°$, AR = 5, 20 Panels Used Spanwise

Therefore, in summary: 20 panels are used in the spanwise direction with a half-cosine spacing, 30 panels are used in the chordwise direction using full cosine spacing. For the wing alone, there are 600 panels.

The planform and airfoil section, previously described, were chosen because this is the experimental setup of Bragg and Khodadoust[47]. In their experiment, Bragg and Khodadoust measured the lift coefficient for the rectangular wing in the Ohio State University wind tunnel at a $R_e = 1.5 \times 10^6$. The experiment involved an angle of attack sweep for a clean wing, i.e. one without an "iced" leading edge, and for a wing with a simulated iced leading edge. The ice shape was taken from tests in the NASA Lewis IRT and is the same shape used in two-dimensional tests of Bragg[20]. Thus, a comparison can be made to both a "clean" wing and an "iced" wing. Figure 21, copied from Reference 45, shows their results. Drag or pitching moment coefficient data is not provided in this report.

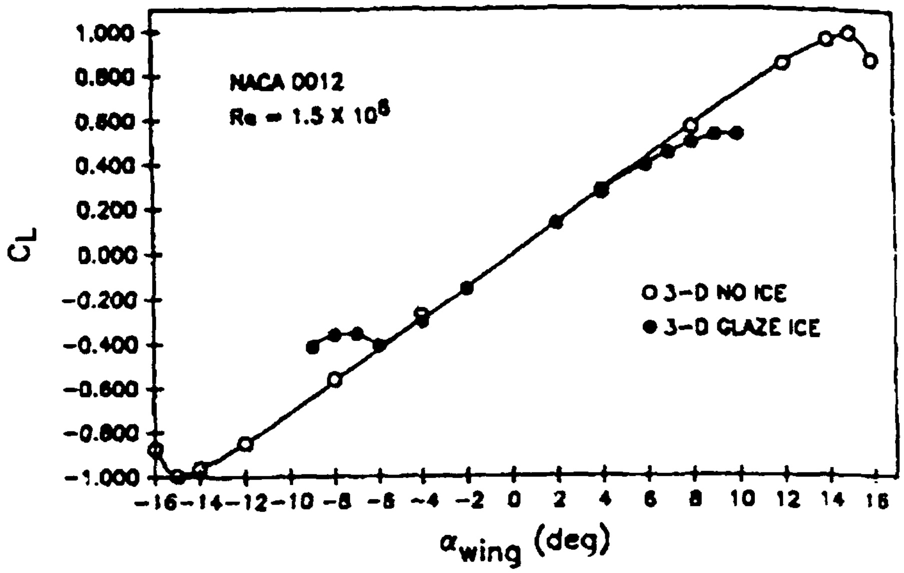

Figure 21. Lift Curve for a Clean Rectangular Wing, AR=5, NACA 0012 Section and with Iced Leading Edge

(Copied from Reference 45)

Initially, an angle of attack sweep was made for the "clean" rectangular wing with viscous corrections. Wake from the wing was allowed to trail parallel to the x-axis. The initial 2-D viscous data used to modify the inviscid solution came from Reference 38. This data provided a

THIS

PAGE

IS

MISSING

IN

ORIGINAL

DOCUMENT

Page 62

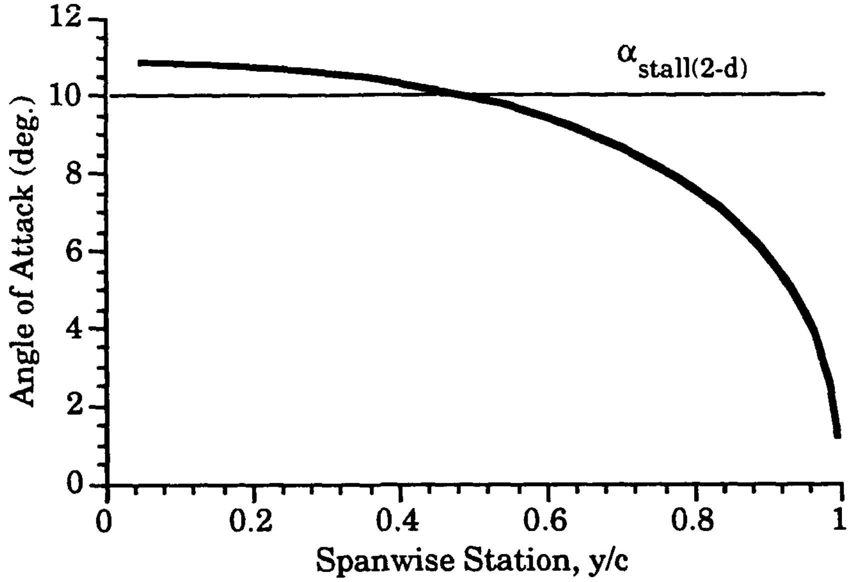

Figure 22. Spanwise Variation of Effective Angle of Attack

Rectangular Wing, $\alpha = 14°$, AR = 5

Note at the inboard stations α_e is greater than $\alpha_{stall(2\text{-}D)}$. Fortunately, the 0012 section is a popular section for study, and Reference 46 provides lift, drag and pitching moment coefficient data for the range $0° < \alpha < 180°$. But this type of "savior" might not always be available, and this problem must be considered when the 2-D data is acquired.

Figure 23 shows a comparison between the experimental lift curve and the computed lift curve of this study with and without viscous effects. PMARC, without viscous effects, greatly overpredicts the lift developed by the rectangular wing. The modifications of this study provide a very good match to the experimental data. Note the accurate estimate of α_{stall}.

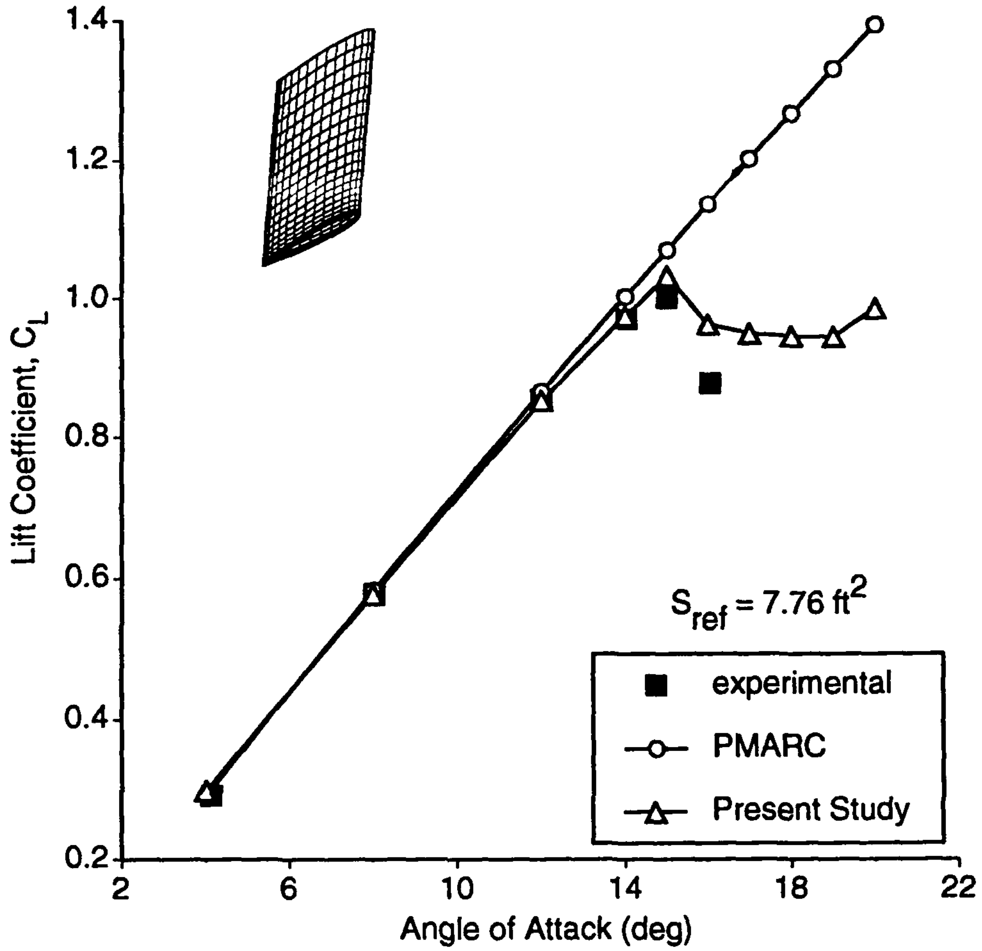

Figure 23. Comparison of Experimental Data for The Clean Rectangular Wing, AR=5, with Values for This Study (Experimental Data From Reference 45)

As a second example of the capability of this method, the previous calculations were reaccomplished using a 2-d dataset for the iced airfoil from Reference 20. Figure 24 shows the results of these calculations. Again, PMARC with the viscous corrections of the current study does a very good job of matching the experimental data. Note that the stall angle of attack is not matched as closely. This can be attributed to the 2-D data which does not include many points past α_{stall}. (See Figure 9, to see how far past stall data is available.) But even with this limited range of 2-D data, the lift curve does show a non-linearity approaching and after α_{stall}. This shows the tendency to provide useful data in the non-linear regime of the lift curve.

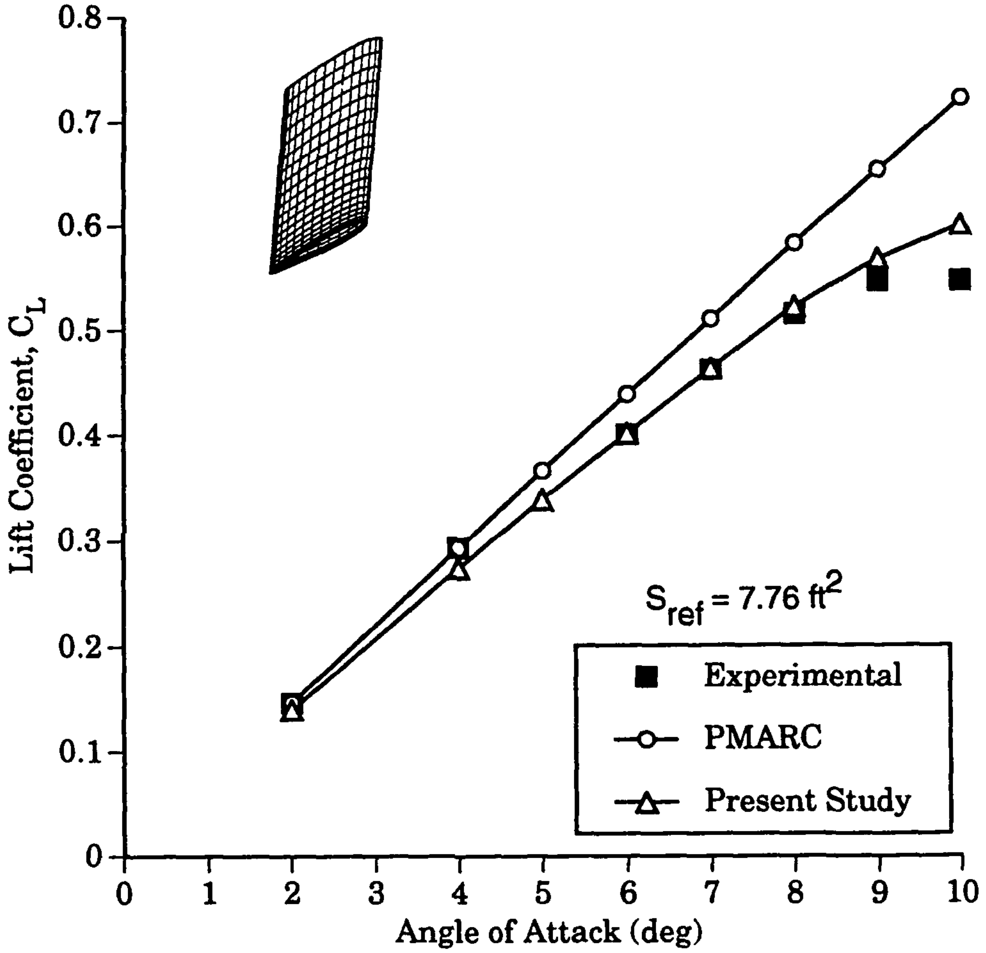

Figure 24. Comparison of Iced Rectangular Wing, AR = 5, with Experimental Data of Reference 45

5.2.2 Wing-Body Results

The next step in the natural progression of this study is to include a fuselage with a wing and make comparisons to experimental data, which will be accomplished in this section.

Figure 25 shows the experimental arrangement for the wing-body configuration. Data computed for this configuration will be compared to experimental results of Reference 47. This model is a generic wing-body configuration with a rectangular wing of a NACA 4412 section, $AR = 8$. All tests in Reference 47 were conducted at a R_e (based on wing chord) of 0.3×10^6. Lift, drag and pitching moment data are provided in the report. Pitching moment is referenced to the half-chord station of the wing.

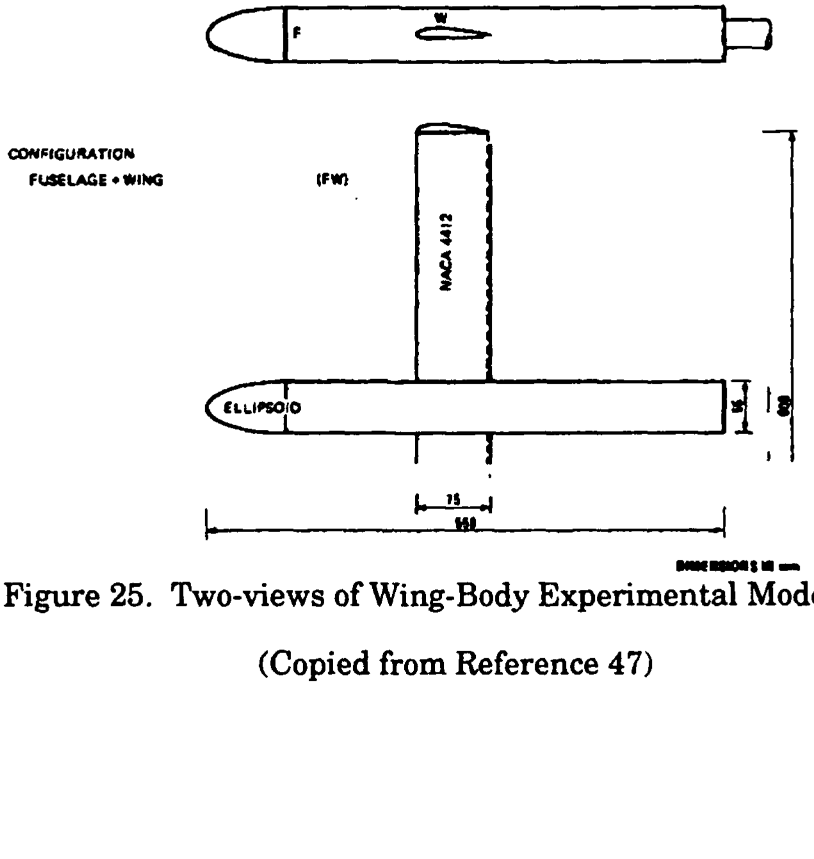

Figure 25. Two-views of Wing-Body Experimental Model

(Copied from Reference 47)

Figure 26 shows the paneling arrangement used in the numerical study of this model. There are 600 panels on the wing and 848 panels for the complete model.

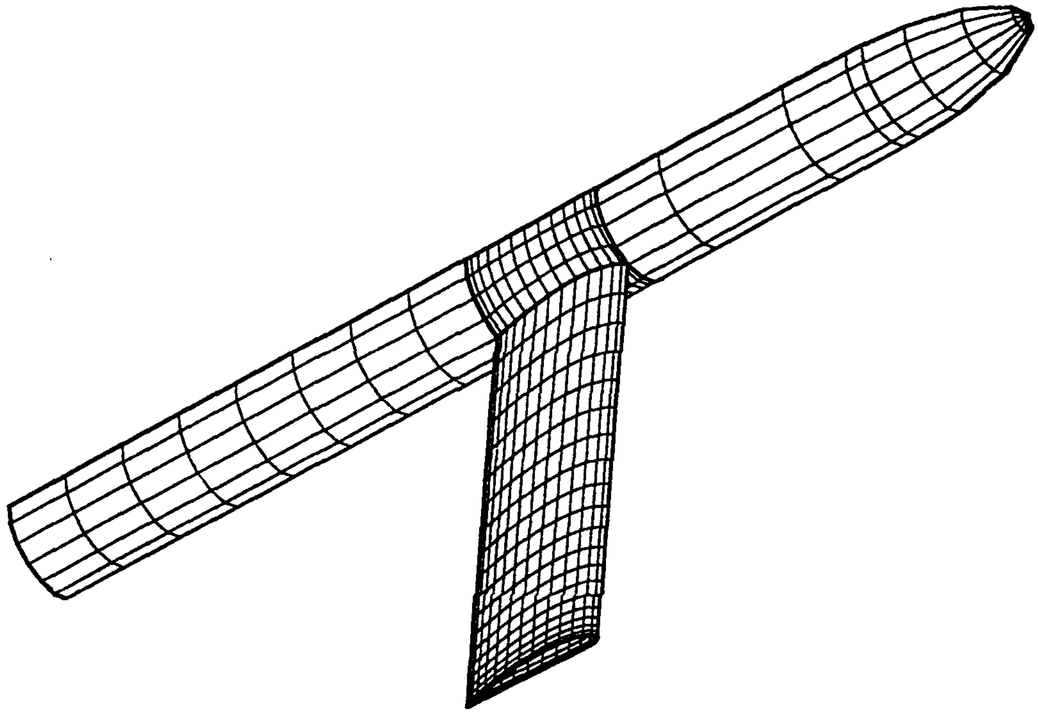

Figure 26. Paneling Layout For The Wing-Body Model

By introducing this new model, two complications are added to the problem. First, there is a new airfoil at a fairly low Reynolds number. Second, the viscous drag of the body is not accounted for.

The first complication is fairly easily satisfied by the airfoil data of Reference 45 which contains wind tunnel data for the NACA 4412 section at a Reynolds number of 0.7×10^6. Unfortunately, the drag and moment coefficients are in standard NACA format, that is they are not presented beyond α_{stall}. This could provide problems near the $\alpha_{stall(3d)}$. The pitching moment data is referenced to the quarter chord of the airfoil section.

The second complication will have to be examined through the calculations. PMARC can model the flowfield changes caused by the body ahead of and behind the wing, but no account is taken for any viscous correction for a non-lifting surface in this method.

An angle of attack sweep was made using the wing-body model. Figure 27a-c presents the results of this sweep for lift, drag and pitching moment.

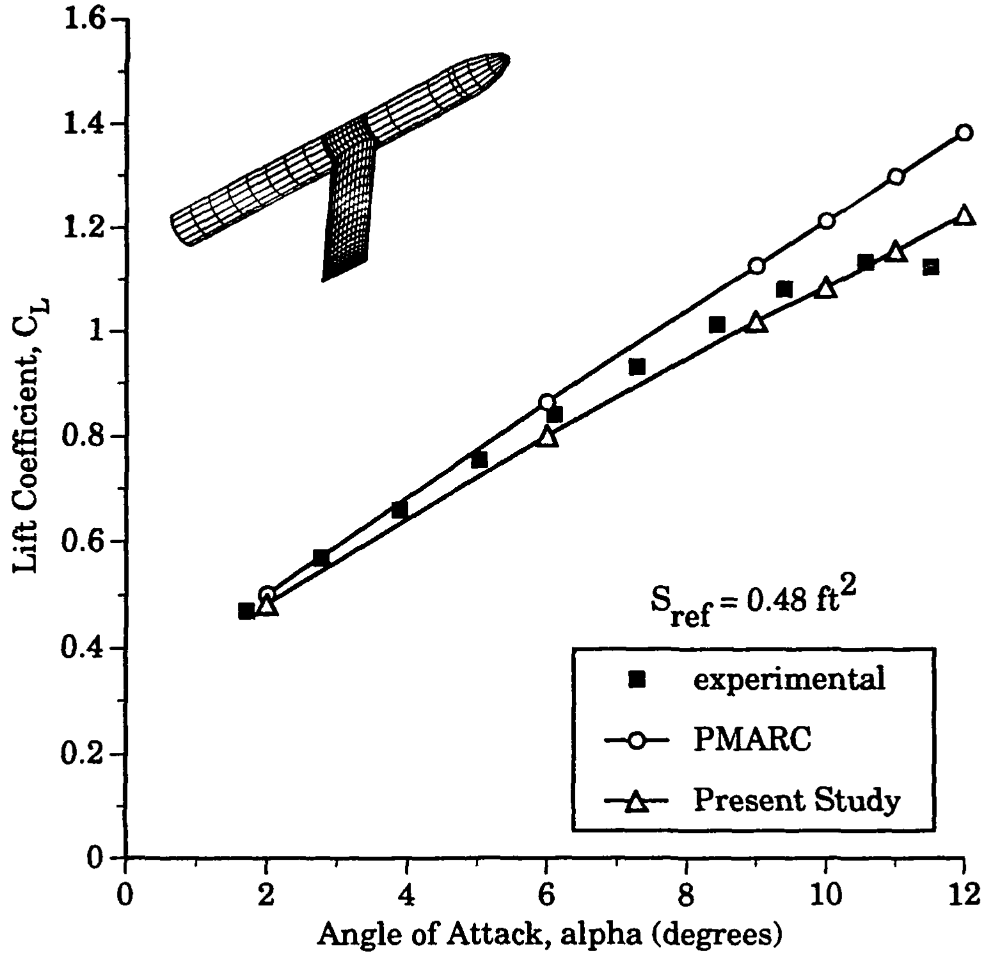

Figure 27a. Lift Curve for Wing-Body Configuration

Experimental Data From Reference 49.

In Figure 27a, as has been the trend from the previous comparisons, PMARC tends to over predict the lift coefficient. The viscous data tends to slightly under predict the experimental results until α approaches α_{stall}. In this region, the viscous data passes through the experimental data, but there is not much of a change in slope, as has been the trend, around α_{stall}. This can be attributed to the 2-D data, as there is not much data presented past α_{stall}. This same effect showed up in the comparison to the iced wing data.

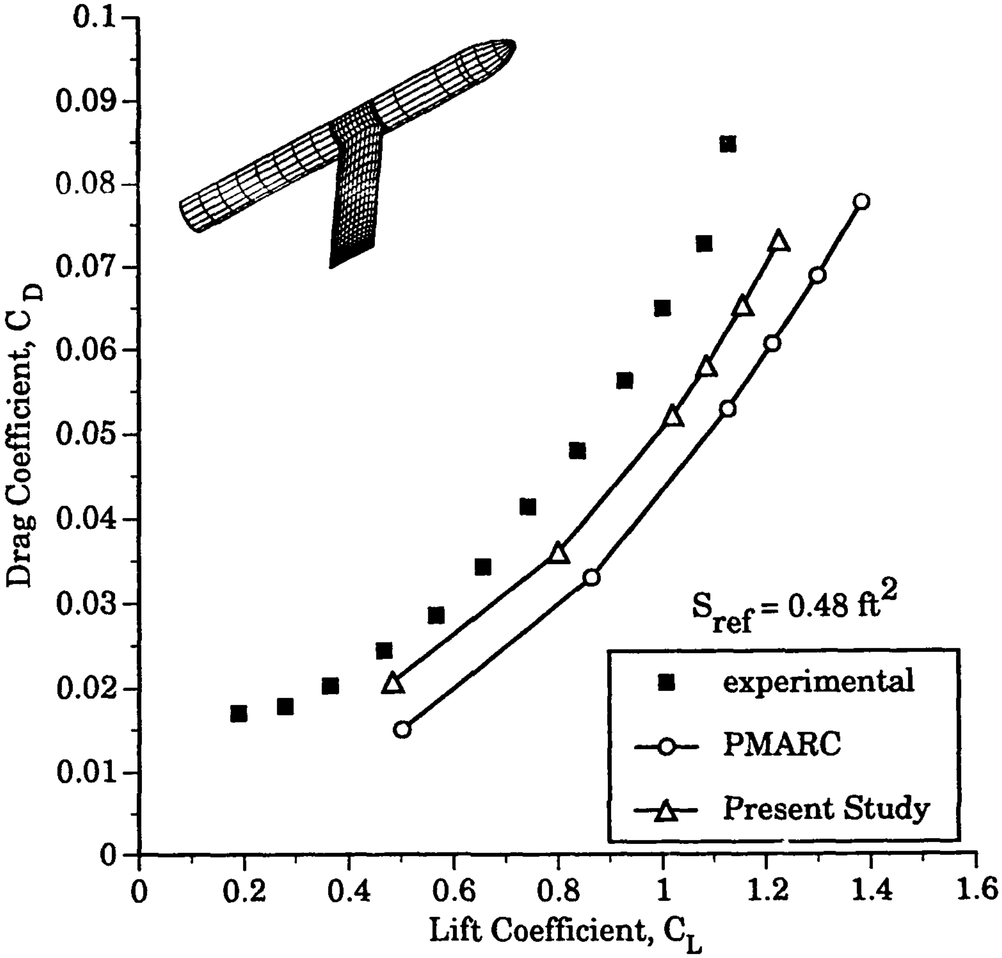

Figure 27b. Drag Polar for Wing-Body Configuration

Experimental Data From Reference 49.

PMARC, for the data of the Present Study in Figure 27b, tends to underpredict the experimental results. Some of this can be attributed to not modeling the viscous drag of the body. But that increment in drag probably would not make up all of the difference. No further explanation is available at this time and will require further investigation.

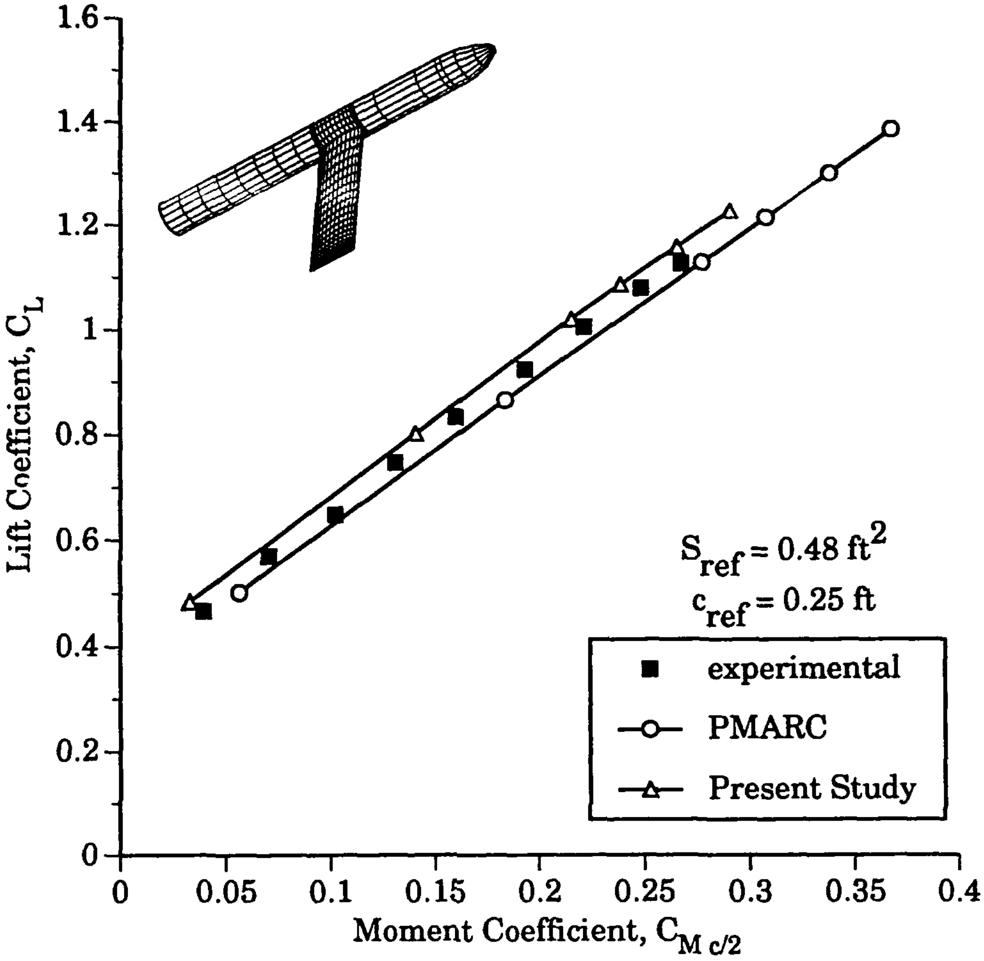

Figure 27c. Moment Curve for Wing-Body Configuration

Experimental Data From Reference 49.

Pitching moment coefficient, in Figure 27c, makes an excellent match of the experimental data. PMARC tended to underpredict the experimental data. This could be expected because the moment caused by the aft portion of the body tends not to model the blunt aft end of the model very well.

5.2.3 Wing-Body-Horizontal Tail Results

The final step in the natural progression of this study is to add a horizontal tail to the wing-body model analyzed in the previous section. This analysis will be accomplished in this section for the full configuration with a "clean" tail and with an "iced" tail. A study showing the sensitivity of wing wake position in relation to the tail will also be presented.

Figure 27 shows the experimental arrangement for the wing-body-low tail configuration. Data computed for this configuration will be compared to the experimental results of Reference 47. The wing and body were described in the previous section. The tail is a rectangular wing of a NACA 0012 section of AR=4.4 mounted on the centerline of the model. Lift, drag and pitching moment data are provided in the report. Pitching moment coefficient is referenced to the half-chord station of the wing.

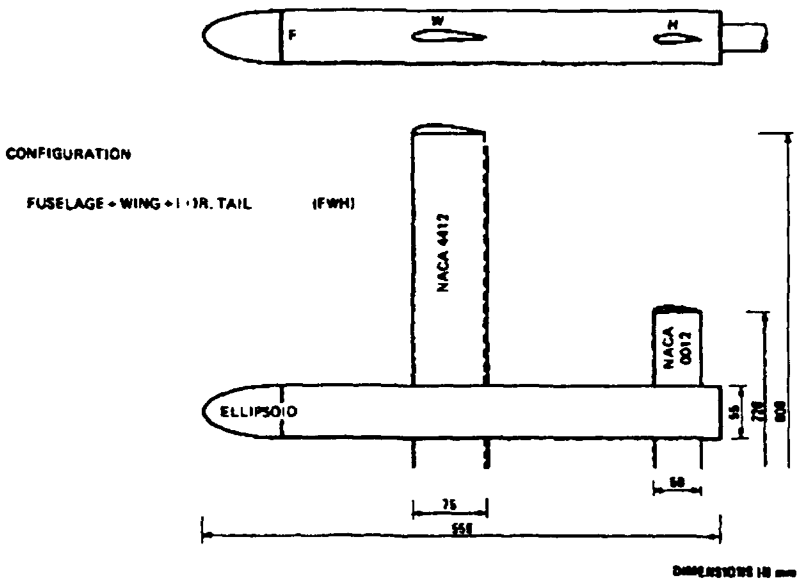

Figure 28. Two-views Of The Wing-Body-Tail Experimental Model (Copied from Ref. 47)

Figure 29 shows the paneling arrangement used in the numerical study of this model. There are 600 panels on the wing, 200 on the tail and 1128 total panels for the complete model.

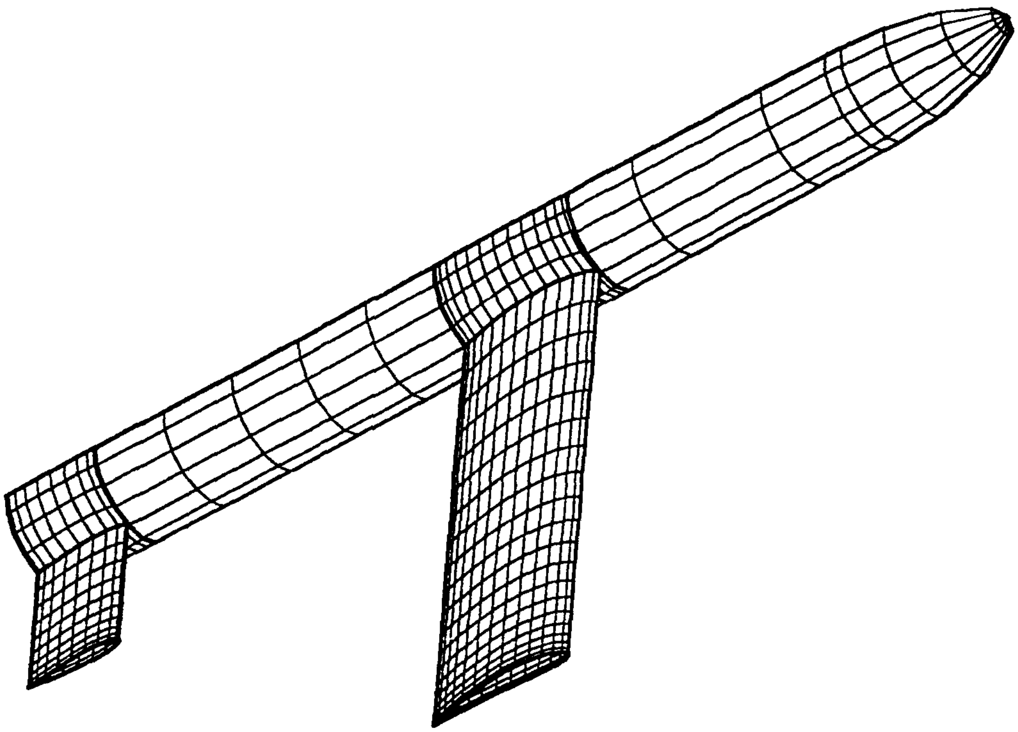

Figure 29. Paneling Layout For the Wing-Body-Low Tail Model

By introducing the coplanar horizontal tail to the model of the previous section, a large complication was added to the problem. Including the coplanar (in the same plane as the wing) horizontal tail adds the problem of "How do you properly model the placement of the wake off the main wing because of the tail location?"

This is a problem because all previous analyses assumed that the wake was parallel to the x-axis. Using this assumption, the wake would cut right through the interior of the horizontal tail. Due to the singularities in the wing wake being on the "wrong" (inside) of the singularities on the tail, computational problems arise. In the development of PMARC (and VSAERO), the internal flow is assumed to be equal to the onset flow when the singularity model is chosen.[34] By putting the wing wake inside of the tail, we are violating this singularity model.

Further research included a discussion with D. Ashby[49], in which he suggested "possibly stitching" the wake to the body under (or over) the tail and adding a short aft cone to the model to bring the wake to the centerline. This was attempted and met with limited to no success. Sectional lift coefficients on the inboard most panel column of the tail were unreasonably high (on the order of 10 or greater) and thus pitching moments were unrealistic. Further attempts at modifying the wake or letting the wake deform met with no success. PMARC does not account for the special case,[34] when the wake is in the plane of the surface panel. So a major modification to PMARC was developed.

This modification involved changing the way the inboardmost wake panel of a lifting surface is handled. The wake is modeled by doublets and the effect of a wake doublet on a surface panel is accounted for through a surface integration The integration considers the effect of a semi-infinite doublet placed on each of the four sides of a wake panel. Due to the way the singularities are placed, PMARC thought the inboard most wake was similar to the wake column at the wingtip, i.e., like a strong tip vortex. The vortex effect is developed because the effect of a surface panel with a doublet is equal to a vortex ring of the same strength as the doublet.[31] To modify PMARC, the value of the integration for the effect of the inboardmost wake column on a surface panel is scaled by the ratio of

$$\frac{\textit{wake panel perimeter} - \textit{length of inmost panel edge}}{\textit{wake panel perimeter}}$$

Using this ratio, is the same as eliminating the inboardmost side of a vortex ring. The effects of the other three sides are canceled by vortices of equal and opposite strength that overlay in the wake mesh. Thus, as is physically realistic, we have removed the vortex along the body and we retain the tip vortex.

The problem of wake placement is mollified, but still not completely removed as the following sensitivity study shows. Figures 30a&b show the results of the sensitivity study for 3 different wake deflections: +1° (which places the wake just above the tail), -1° (which places the wake just below the tail) and +3° (for symmetry about the +1° data point).

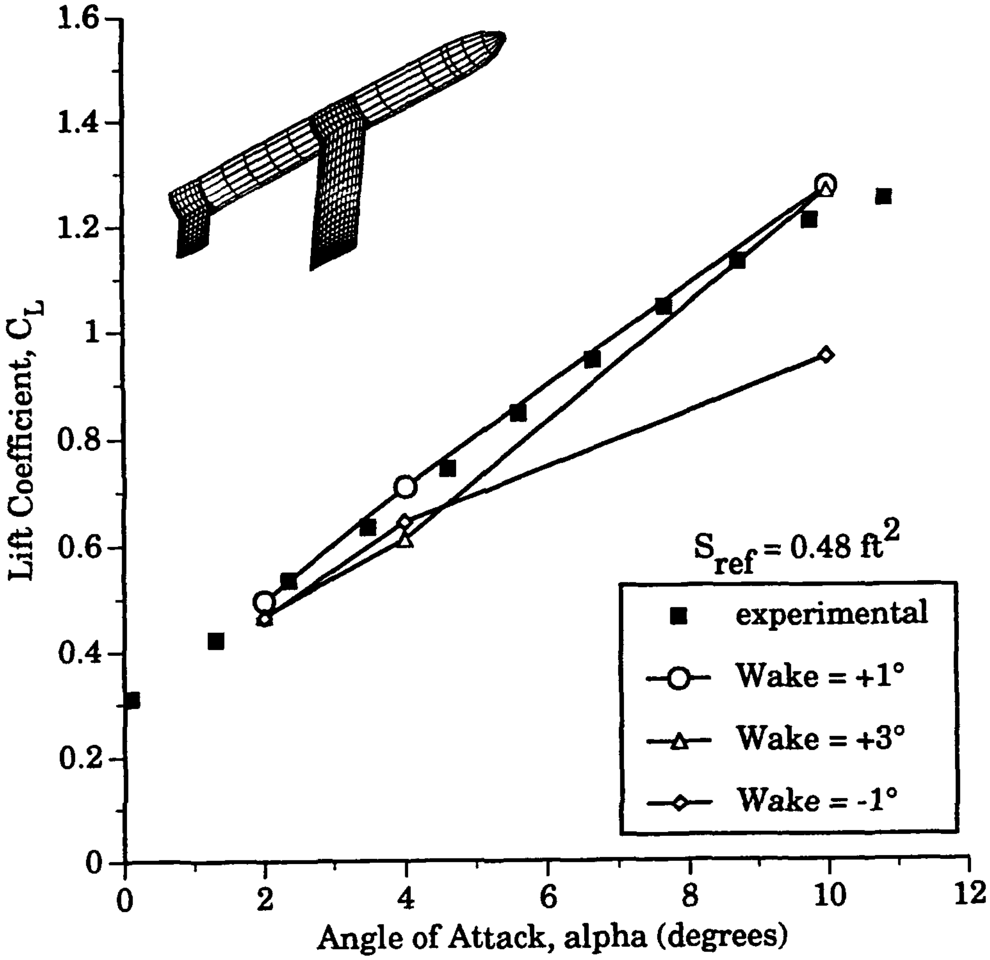

Figure 30a. Wake Position Sensitivity on Lift Coefficient,

Wing-Body-Low Tail Configuration

(Experimental data is from Reference 47)

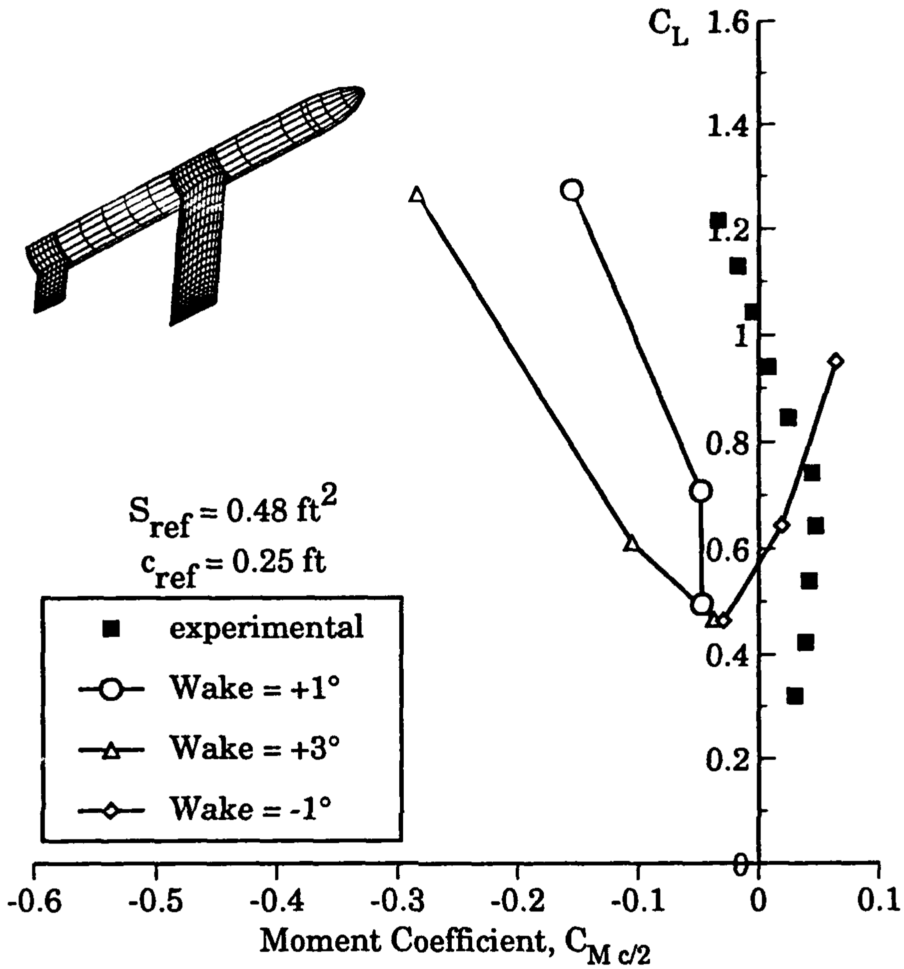

Figure 30b. Wake Position Sensitivity on Pitching Moment Coefficient, Wing-Body-Low Tail Configuration
(Experimental data is from Reference 49)

As Figures 30a & b demonstrate, the assumed position of the wake has a strong effect on the pitching moment. Putting the wake under the tail, causes large negative lift on the tail and a strong pitchup (positive) moment. Putting the wake relatively high above the tail causes a large pitchdown moment. Putting the wake as close as is possible to the x-axis appears to be the best solution. Because of the coplanar tail, the

+1°-deflection is a good compromise. This is the assumed wake deflection throughout the rest of this study. Note that the wake position has little effect on the global lift coefficient at lower angles of attack, but the location of the wake relative to the tail has a large effect at higher angles of attack.

Using the results of this sensitivity study, the full configuration can be analyzed. Figures 31a-c show plots of the longitudinal force and moment coefficients for the full configuration with a low tail.

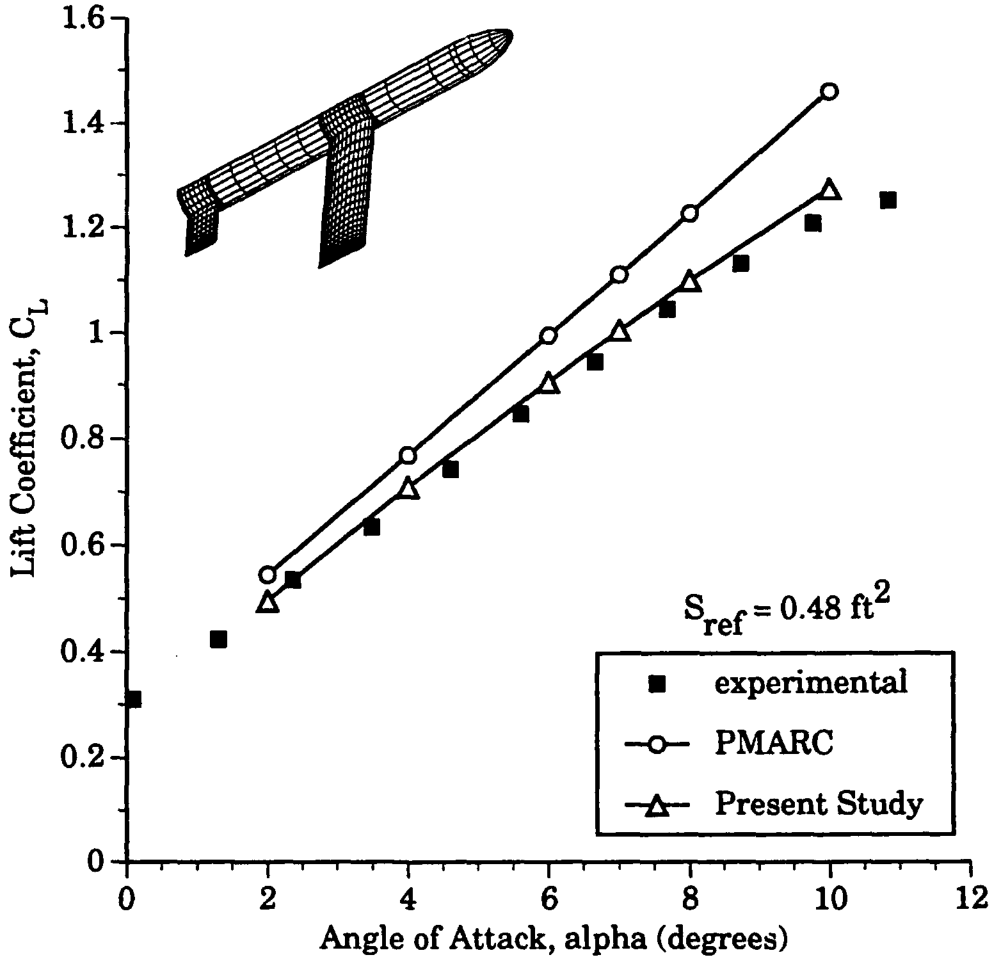

Figure 31a. Lift Curves for Wing-Body-Low Tail Configuration (Experimental Data From Reference 47)

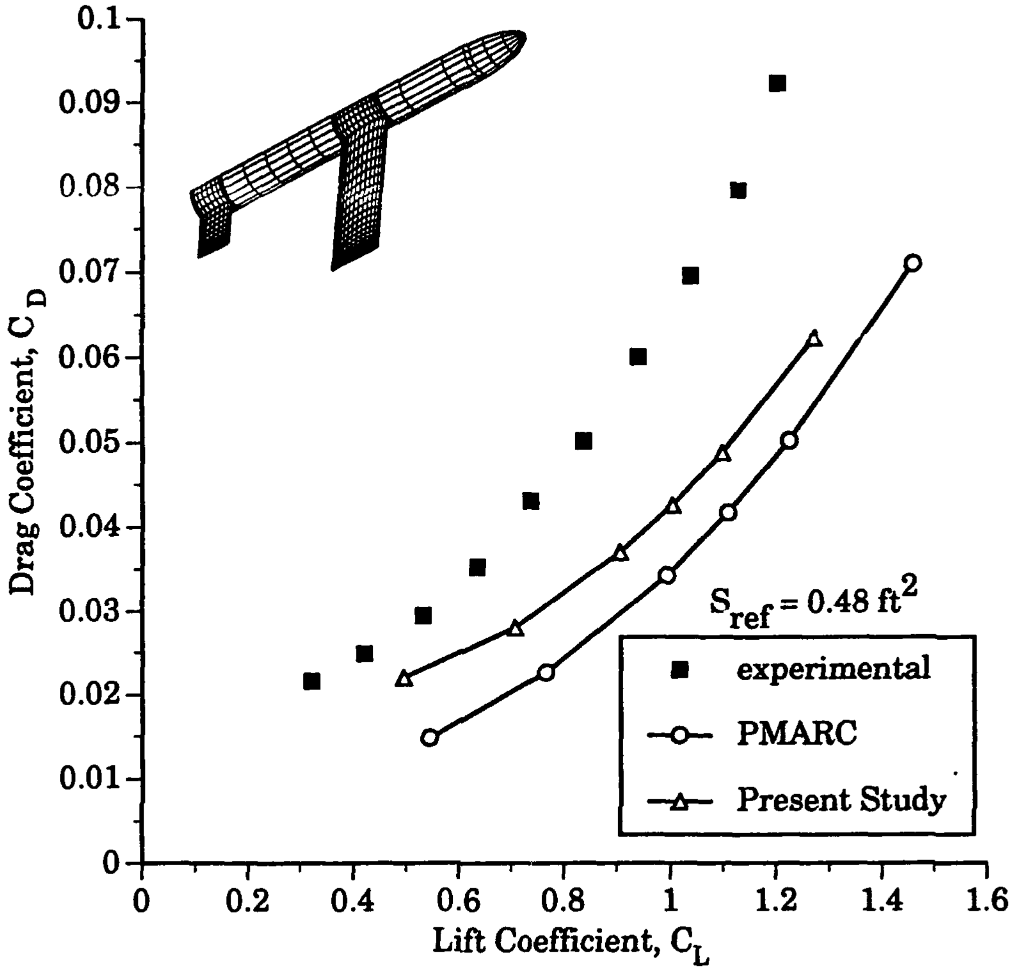

Figure 31b. Drag Polar for Wing-Body-Low Tail Configuration

(Experimental Data From Reference 47)

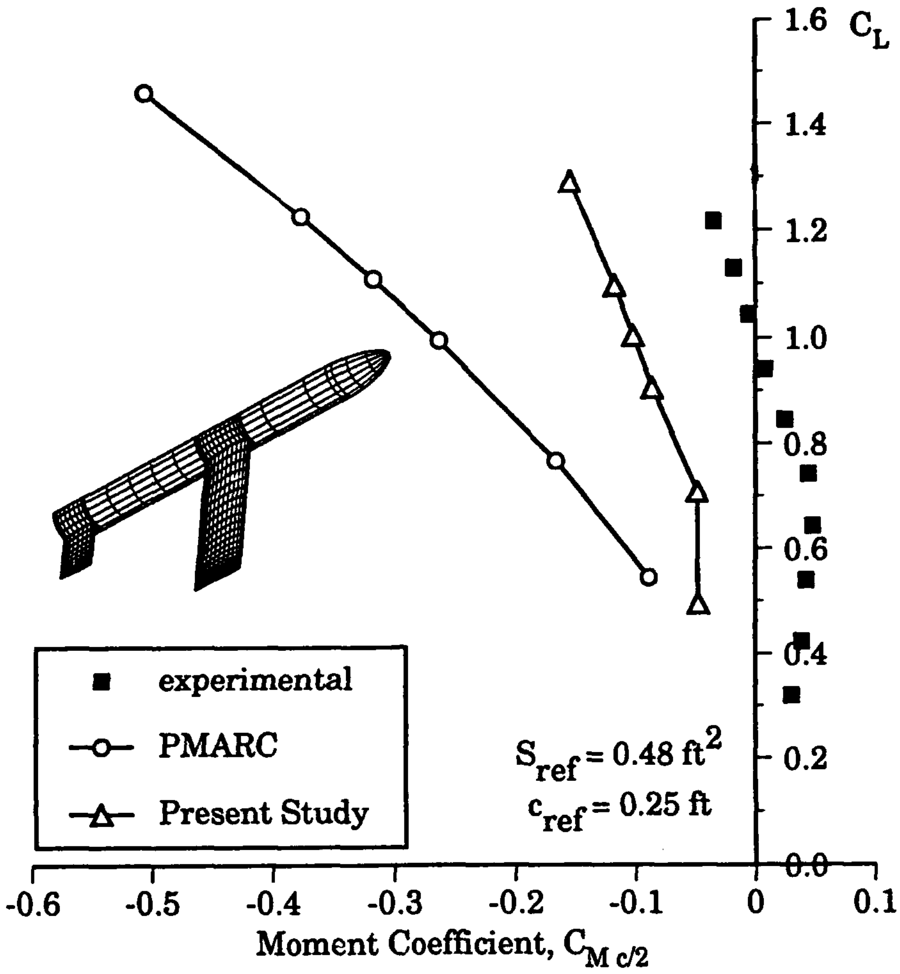

Figure 31c. Pitching Moment Curve for Wing-Body-Low Tail Configuration

(Experimental Data From Reference 47)

In Figure 31a, one can see that the viscous modifications to PMARC do an excellent job of matching the experimental lift coefficient. Remember, viscous corrections are made to each lifting surface, but not to the body. The drag coefficient again is underpredicted, in Figure 31b, but is an improvement over the inviscid solution. The pitching moment of Figure 31c follows the same trend as the experimental data, but is displaced in a more stable direction. This can partly be attributed to the wake deflection problem.

Using this same configuration, one can now consider that the tail has accumulated ice along the leading edge. To model this, the leading edge of the main wing doesn't need to have accumulated any ice as Reference 46 has stated. Furthermore, as Reference 50 stated, "The horizontal tail will accumulate ice up to four times as fast as the main wing due to the usually smaller cross-sectional area and smaller radius of curvature than the main wing." Thus, this study will model this situation with ice only on the tail.

To model this new situation all that has to be changed is the 2-D data input set for the tail, i.e. replace the viscous 2-D clean airfoil data with viscous 2-D <u>iced</u> airfoil data (obtained either experimentally or analytically). Figures 32a-c show the results of this analysis.

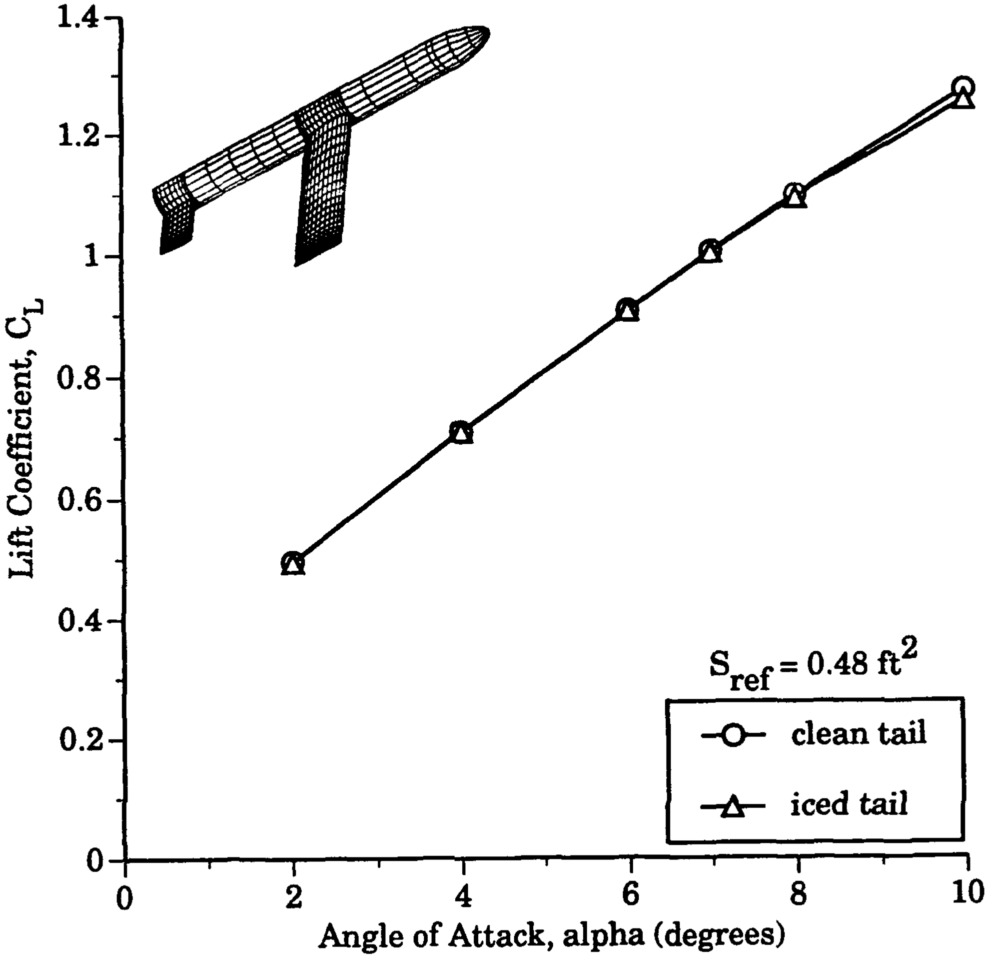

Figure 32a. Lift Curve for Wing-Body-Tail Configuration with Iced Low Tail

(Experimental data is from Reference 47)

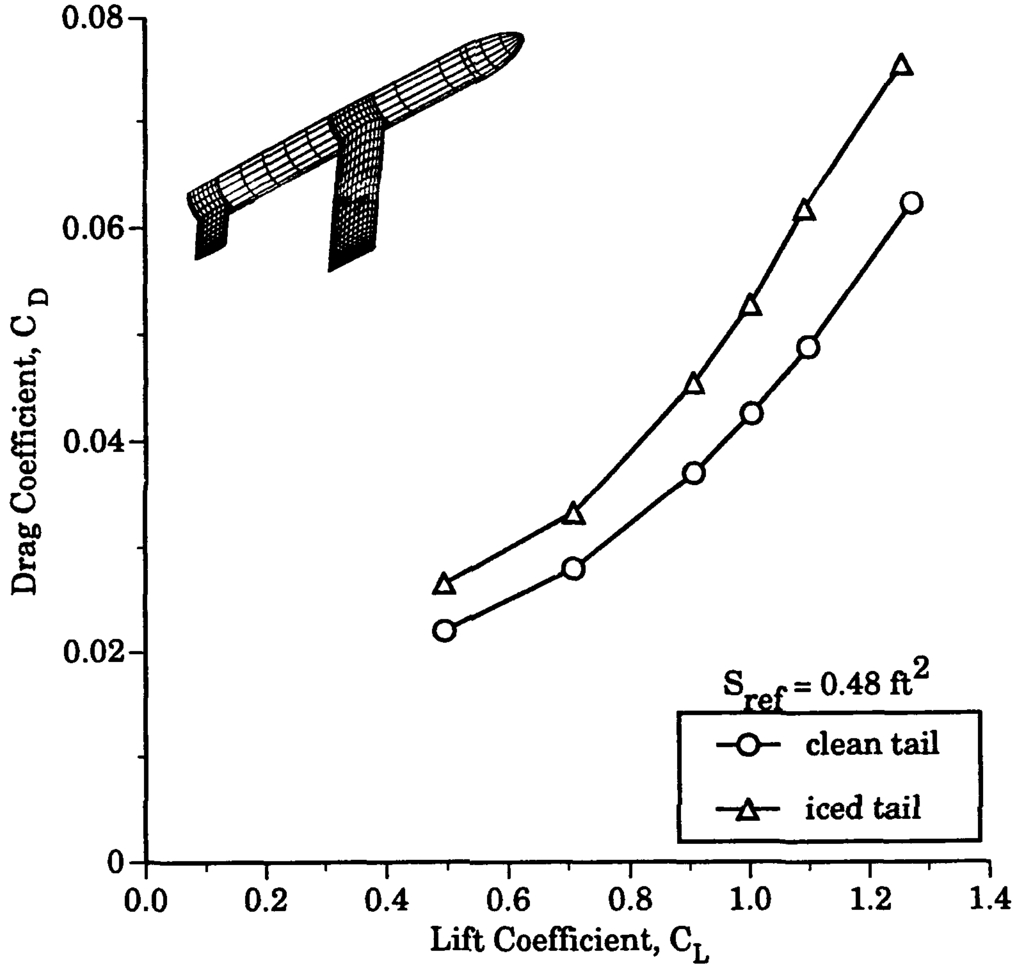

Figure 32b. Drag Polar for Wing-Body-Tail Configuration with Iced Low Tail

(Experimental data is from Reference 47)

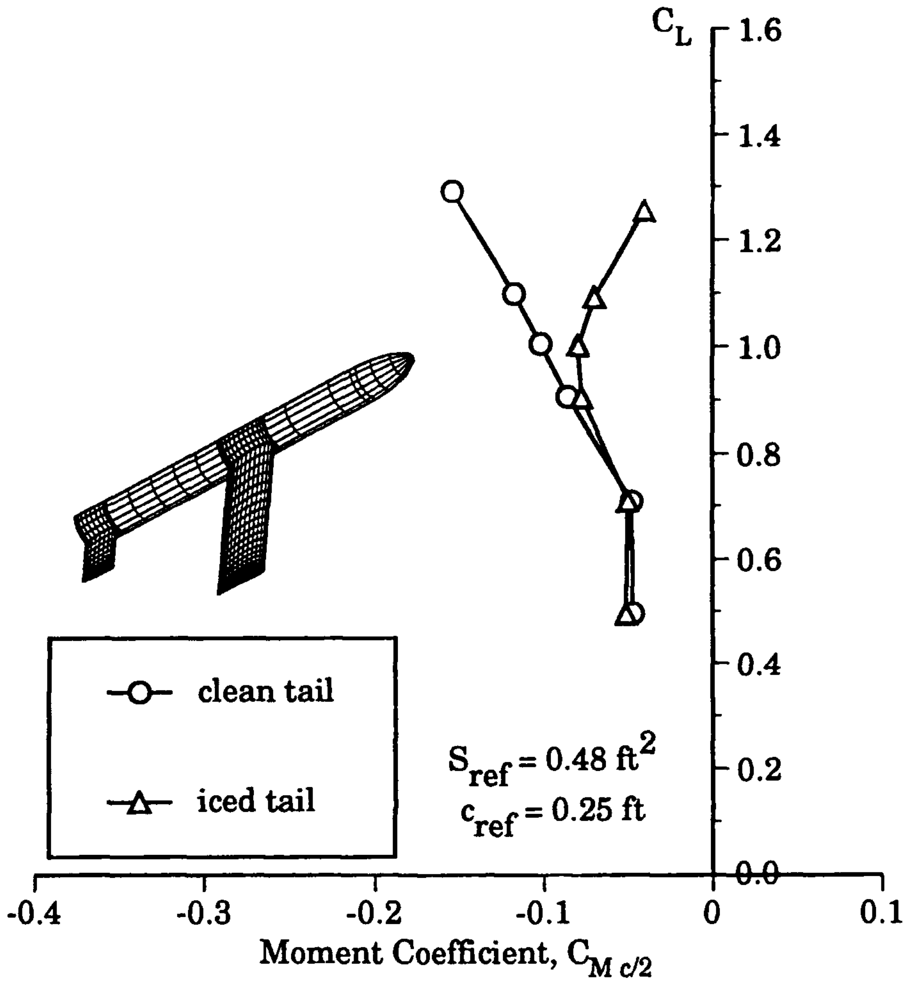

Figure 32c. Pitching Moment Curve for Wing-Body-Tail Configuration with Iced Low Tail

(Experimental data is from Reference 47)

Figure 32a shows there is very little change in the lift coefficient with ice on the tail. (Compare this to Figure 3 at $\alpha < 5°$.) Notice at $\alpha = 10°$, C_L for the iced configuration is less than for the clean configuration. This shows that some portions of the iced tail are approaching α_{stall}. (Figure 3 shows a large difference between c_l for the clean airfoil vs. the iced airfoil at α_{stall} for the iced airfoil.) The drag coefficient is seen to increase as is expected. Figure 31c shows that the biggest change with ice included is in the pitching moment. Though the values of the pitching moment don't match the experimental values, the trends are important. Initially, at low lift coefficients, the ice has little effect. As the lift coefficient increases, one sees a slight reduction in the stability. Then, as discussed in Reference 10, one sees a prediction of a dramatically unstable pitch break. (It would be dramatic from the pilot's point of view.)

Physically, what is happening to cause this unstable pitch break? Looking at Figure 3, one can see that there is little difference in lift coefficients at lower angles of attack. But as the angle of attack is increased, this difference is increasing rapidly. Because the horizontal tail provides a small portion of the total lift coefficient, one would not expect to see a big change in total lift coefficient as angle of attack increases. This is what Figure 32a shows. Now think about the total pitching moment. This change in sectional lift coefficient with change in angle of attack will have a large effect as the angle of attack increases. And at some angle of attack, the lift on the tail will start to decrease as parts of the tail stall with increasing angle of attack. This causes the pitch break that is demonstrated in Figure 32c. (This explanation shows why the "similar attempts" didn't work. You can not just apply the 2-D values on the model. You need the full 3-D effect, which is included by matching the local circulation.)

One final study will examine the effects of moving tail out of the downwash of the wing. As a secondary effect, the wing wake placement is no longer interfering with the tail. Though the wake could parallel the x-axis, it will be modeled with the same deflection as the previous study to eliminate this variable from analysis of the results. In this study, the tail is moved $1.7\bar{c}$ above the previous tail location. Figure 33 shows the paneled model for this study.

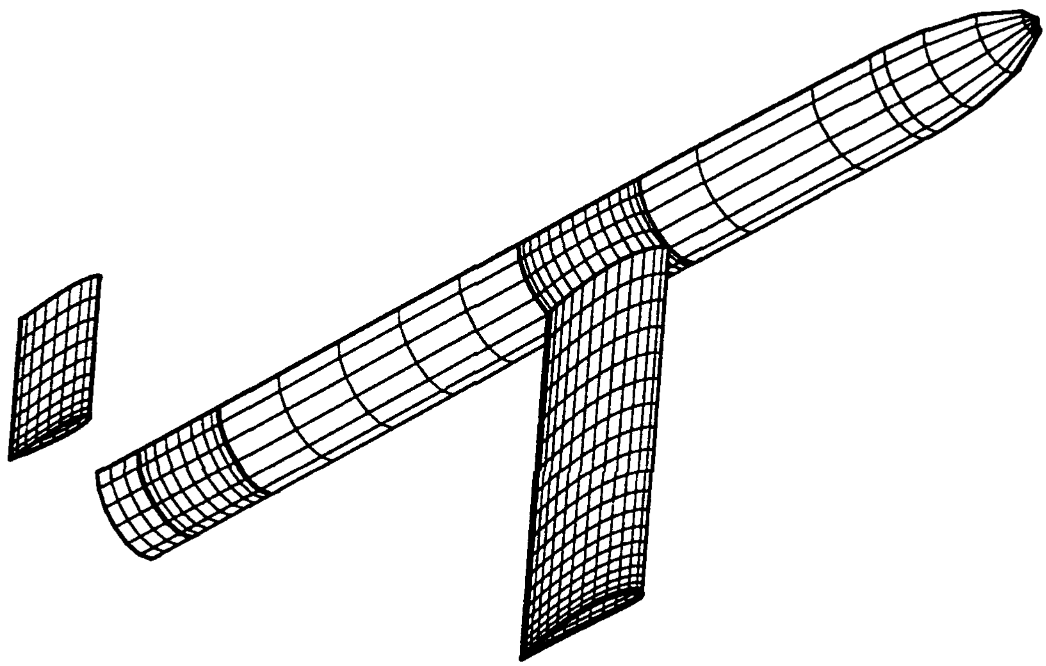

Figure 33. Paneling Layout for Wing-Body-T-tail Configuration

Figures 34a-c show the results for a wing-body-t-tail (wbt-t) configuration with a clean tail and with an iced tail.

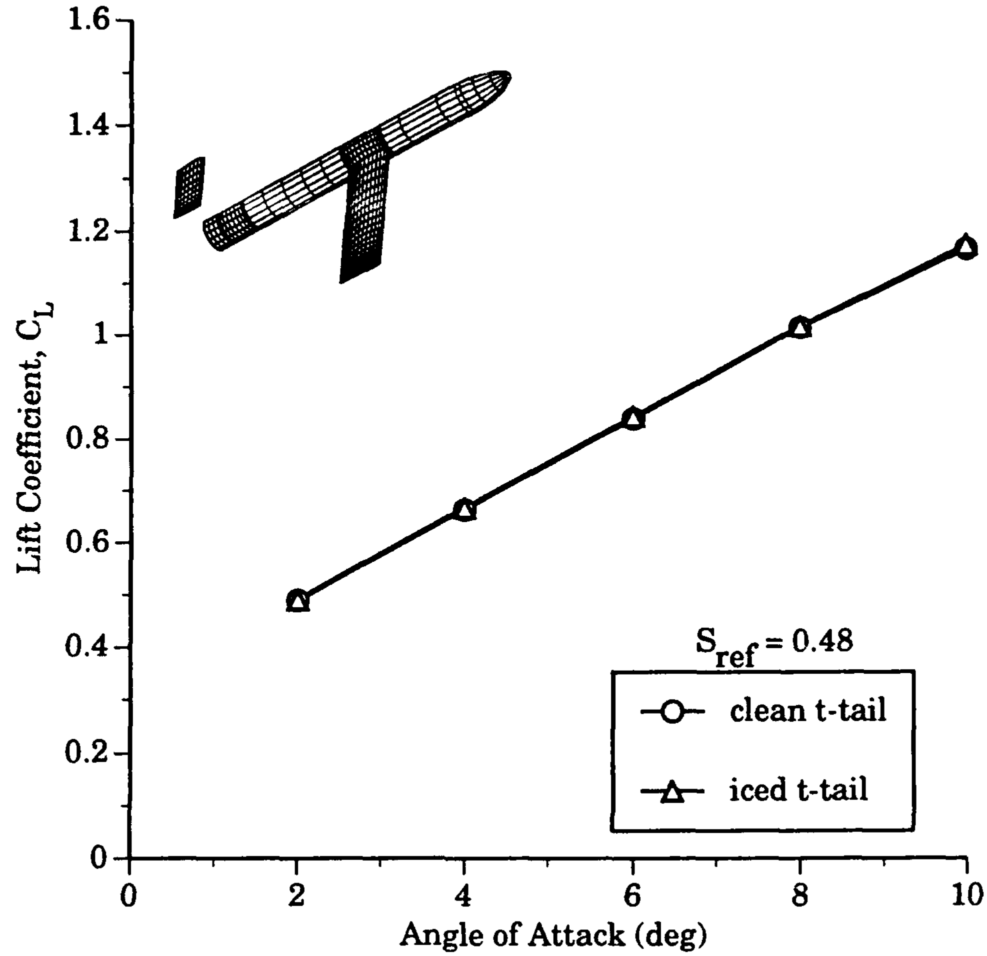

Figure 34a. Lift Curve For The WBT-T Configuration

Figure 34a follows the trend of the previous study: there is very little if any difference in lift coefficient between the clean configuration and the iced configuration. This is due to two factors:

(1) the size difference between the two lifting surfaces, i.e. the tail provides a much smaller portion of the lift;

(2) the spanloading of the tail for the iced configuration. Not all of the tail is in a regime where the lift has started to reduce with increasing α, thus the overall lift does not change much.

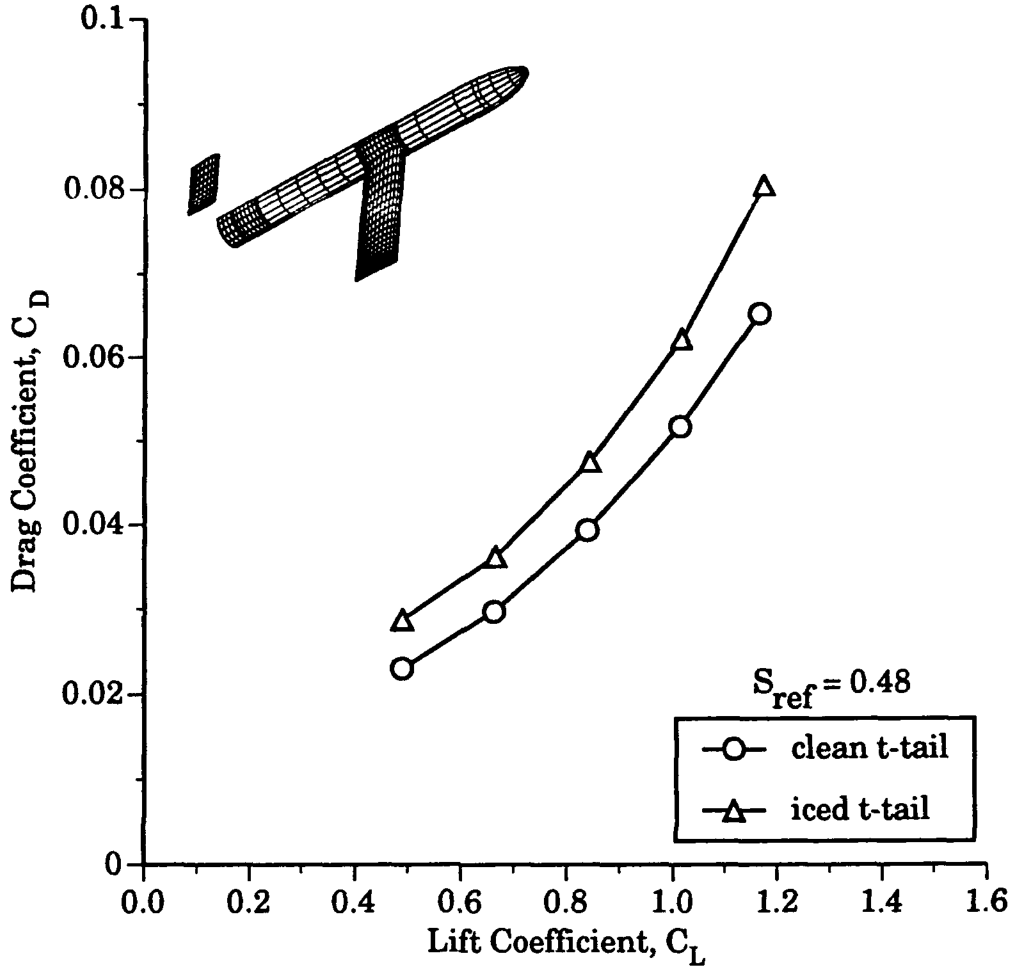

Figure 34b. Drag Polar For The WBT-T Configuration

Figure 34b shows the drag polar for the clean and iced t-tail configuration. This figure follows the expected trend of an increase in drag coefficient for the iced configuration over the clean configuration.

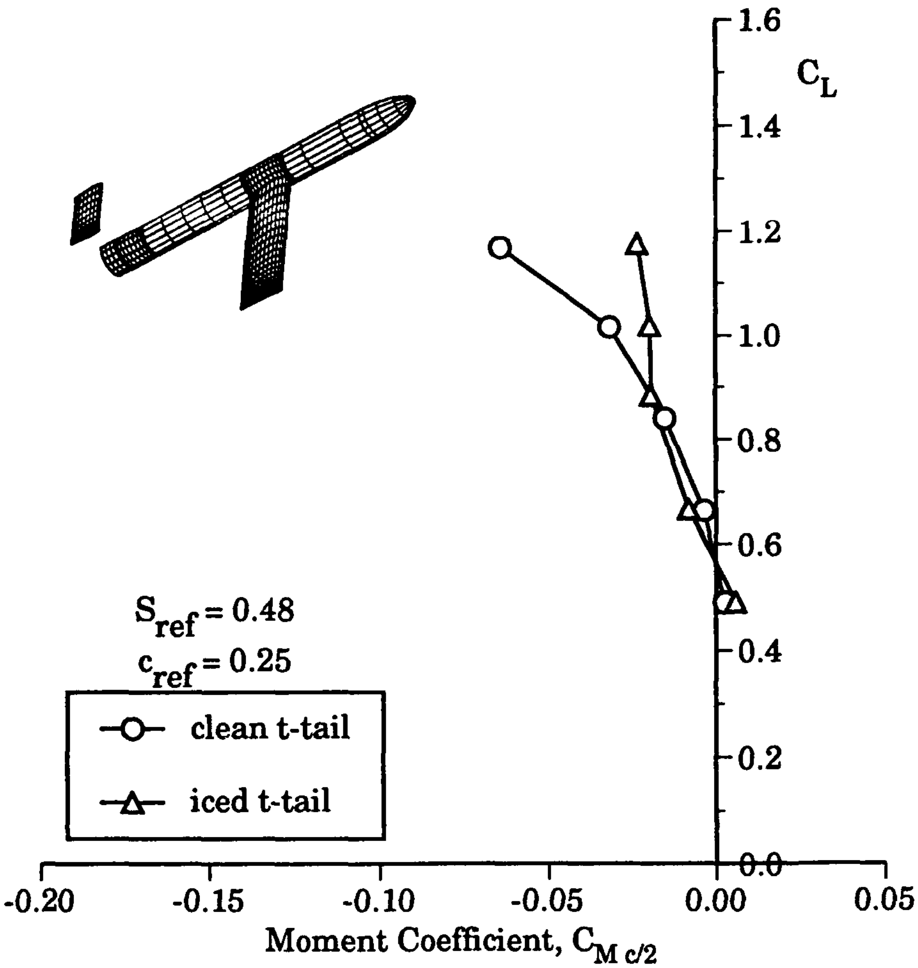

Figure 34c. Pitching Moment Curve For The WBT-T Configuration

Figure 34c shows the pitching moment coefficient for the wbt-t configuration. This plot shows a very interesting development--the pitching moment starts out to be positive as would be expected from the experimental study with a low tail configuration. As the lift coefficient increases to about 0.7, the stability increases for both conditions. Then, while the clean configuration becomes more stable, the pitch stability decreases for iced configuration. This change in stability condition for the iced configuration can be explained by two factors with reference to Figure 3. The first deals with the reduction in lift for an iced airfoil. Remember, as the angle of attack increases, the lift coefficient for the iced airfoil does not increase as quickly as for the clean airfoil. To put it another way, $c_{\ell_{\alpha(clean)}} > c_{\ell_{\alpha(iced)}}$. The second factor involves the effect of ice on airfoil pitching moment. (Again, refer to Figure 3.) For the iced airfoil, we see a large decrease in pitching moment with increasing c_ℓ above $c_\ell \approx 0.4$. This pitching moment trend provides a moderating factor for the decreasing c_ℓ at higher α's. These two factors explain the change to neutral stability in Figure 34c for the iced configuration above $C_L \approx 0.8$ compared to the unstable response of Figure 32c.

Figure 35 shows the capability of combining the results of 2-D calculations with 3-D calculations to understand the complicated flow phenomena involved without having to solve for the total flow field.

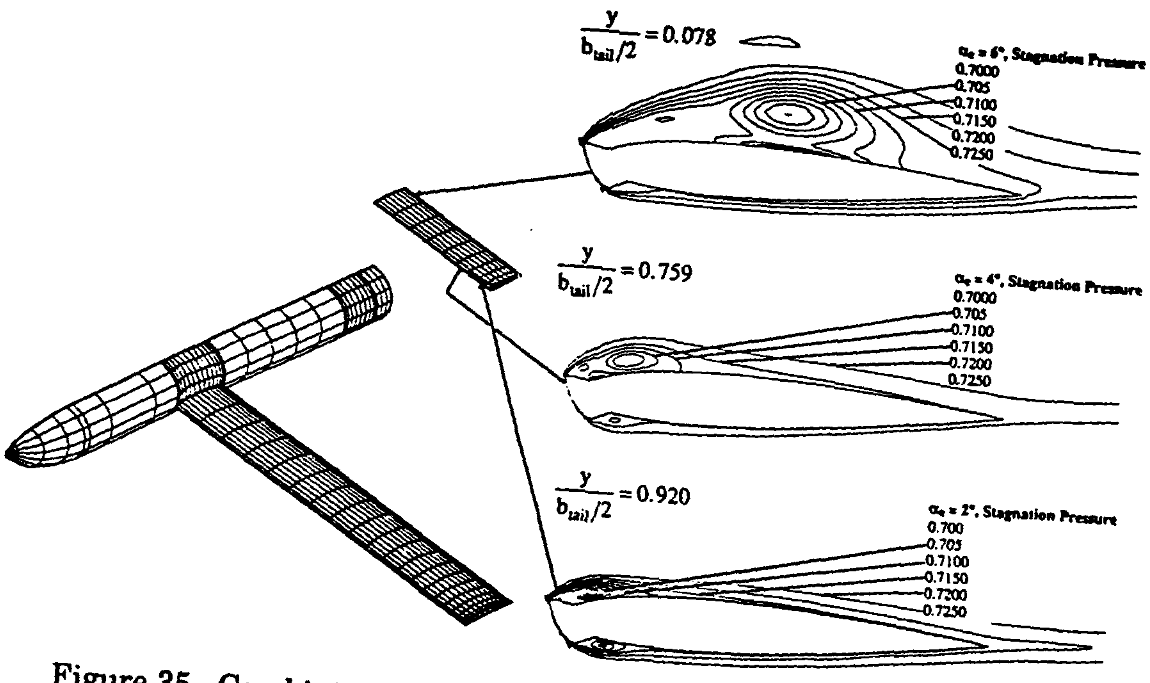

Figure 35. Combining Flow Field Calculations of Modified PMARC and ARC2D, α=10°, (Total Pressure Contours)

Figure 35 shows the power of this method. One of the values used to modify the circulation on the tail is the effective angle of attack, α_e. Combining the spanwise variation in α_e with the 2-D calculations about the viscous airfoil, one can get a feel for what is happening spanwise across the tail with simulated ice. For example, from Figure 35 one can see as the effective angle of attack increases, the vortex above the wing grows. (Compare the PLOT3D[42] total pressure contours at the inboard and outboard stations.) From experimental studies[20], it has been shown that this vortex is shed periodically. Thus the inboard portion of the tail would demonstrate an unsteady nature. Notice also how little the flow field at the outboard station of the tail is affected by this vortex,. This picture helps to explain the small difference in lift coefficients between clean and iced configurations at the lower geometrice angles of attack.

Chapter 6
Conclusions and Recommendations

This chapter lists conclusions from the present study along with recommendations for future work.

6.1 Conclusions

A new method has been developed to quickly analyze a full aircraft configuration taking into account viscous effects. Experimental sectional data or precomputed viscous 2-D calculations, from a code like ARC2D, were used as input to the modified panel code PMARC. PMARC has been modified to find an iterated solution which matches the local circulation. From this matching condition C_L,, C_D and C_M for the complete configuration were calculated that include viscous effects for lifting surfaces.

Using this modified version of PMARC and appropriate 2-d viscous sectional data, it has been shown that:

(1) The lift curve slope has been accurately calculated for a wing alone. If 2-D sectional data were available for a high enough angle of attack, α_{stall} could be accurately calculated.

(2) The lift and pitching moment curves for a wing-body configuration have been accurately calculated. There was some discrepancy in the drag polar. Part of this discrepancy could be accounted for because viscous effects for the body were not included.

(3) The lift curve for two wing-body-tail configurations have been accurately calculated within the constraints of the 2-D data available. Drag polar and pitching moment curves exhibit the proper trends, but

did not match the experimental data as well. The pitching moment data did predict the reduction in stability due to icing effects on the tail.

6.2 Recommendations

The viscous analysis of flow over an airfoil past stall does not model the physical situation very well. Indications were that the turbulence model used in ARC2D does not properly model the physics. Further investigation into a proper way to model the effect of turbulence with a simple, efficient model would improve the capability of ARC2D and greatly improve the capability of the method of the present study.

Because Reynolds number has a great effect on the 2-D flowfield about an airfoil near stall, Reynolds number effects need to be included in this modification to PMARC to make it more widely applicable.

Further investigation into the modeling of the wake shed by the lifting surfaces would ease the use of PMARC. Currently the position of the wake has a great effect on the results when the wake is near a lifting surface and yet with certain configurations this situation can not be avoided. Especially in the coplanar tail model studies, the wake position had a large effect on pitching moment coefficient. Thus a better way to model the wake is necessary to correct the magnitude of the pitching moment coefficient.

One area that has not been considered but is a natural outshoot of this research is the effect of icing on the elevator hinge moment. Ice on the tail causes the flow field over the elevator to be distorted. Due to this distorted flow field, hinge moments can change. These changes affect the pilot's recovery capability with an iced tail. Being able to predict the hinge moments would facilitate control system design and recovery techniques with an iced tail.

Chapter 7

References

1. "Aircraft Icing Avoidance and Protection," NTSB-SR-81-1, National Transportation Safety Board, Washington, D.C., 20594.

2. "Annual Review of Aircraft Accident Data, U.S. General Aviation Calendar Year 1978," NTSB-ARG-80-1, National Transportation Safety Board, Washington, D.C., 10594.

3. "Annual Review of Aircraft Accident Data, U.S. General Aviation Calendar Year 1980," NTSB/ARG-84/01, National Transportation Safety Board, Washington, D.C., 10594.

4. "Annual Review of Aircraft Accident Data, U.S. General Aviation Calendar Year 1984," NTSB/ARG-87/02, National Transportation Safety Board, Washington, D.C., 10594.

5. "Aircraft Accident Report, Ozark Air lines, Inc., DC-9-15, Sioux City Airport, Sioux City, Iowa, December 27, 1968." NTSB-AAR-70-20, Washington, D.C. 20591.

6. "NTSB Seeks Cause in Crash of 737", Aviation Week and Space Technology, January 18, 1982.

7. Excerpts from the NTSB Accident Report on the Air Florida 737 Crash, Aviation Week and Space Technology, October 4, 1982, October 18, 1982 and November 22, 1982.

8. "Crash Investigators Seek Clues in F-28's Major Systems," Aviation Week and Space Technology, March 30, 1992.

9. "Icing Procedures Under Investigation," Flight International, 1-7 April, 1992.

10. Dow, J. P., Sr., Lium, G. D., and Perkins, P. J., "Tailplane Stall, The Rime Is The Reason", NBAA Aviation Issues, Feb, 1993.

11. "Airworthiness Standards: Normal, Utility, Acrobatic, and Commuter Category Airplanes", Code of Federal Regulations, Title 14, Pt. 23, 1991.

12. "Airworthiness Standards: Transport Category Airplanes", Code of Federal Regulations, Title 14, Pt. 25, 1991.

13. Ferarrio, F. and Wallis, T. E., "SF 600 'Kangaroo' Tanker Development and Icing Flight Test Results", 23rd Annual Symposium Proceedings, Society of Flight Test Engineers, August 3-7, 1992.

14. Laschka, B. and Jesse, R. E., "Ice Accretion and its Effects on Aerodynamics of Unprotected Aircraft Components," AGARD Advisory Report 127, Nov. 1978.

15. Reinmann, J.J., Shaw, R.J., and Ranaudo, R.J., "NASA's Program on Icing Research and Technology," NASA TM 101989, Symposium on Flight in Adverse Environmental Conditions, Gol, Norway, May 8-12, 1989.

16. Bragg, M. B. and Gregorek, G. M., "Predicting Aircraft Performance Degradation Due to Ice Accretion," SAE Paper 830742, Business Aircraft Meeting and Exposition, Wichita, KS, April 12 15, 1983.

17. Potapczuk, M. G., Bidwell, C. S., and Berkowitz, B. M., "Progress Toward the Development of an Airfoil Icing Analysis Capability," NASA Computational Fluid Dynamics Conference, Moffett Field, CA, March 7-9, 1989.

18. Potapczuk, M. G. and Bidwell, C. S., "Swept Wing Ice Accretion Modeling," AIAA 90-0756, 28th Aerospace Sciences Meeting, Reno, NV, January 8-11, 1990.

19. Reinman, J. J., "NASA's Aircraft Icing Technology Program," NASA-TM-104518, 1991 Winter Annual Meeting of the ASME, Atlanta, GA, 1-6 Dec 1991.

20. Bragg, M. B., "Experimental Aerodynamic Characteristics of an NACA 0012 airfoil with Simulated Glaze Ice," Journal of Aircraft, Vol. 25, No. 9, September, 1988.

21. Bidwell, C. S., "Icing Characteristic of a Natural-Laminar-Flow, a Medium-Speed, and a Swept, Medium-Speed Airfoil," NASA TM 103693, January, 1991.

22. Bragg, M. B., Gregorek, G. M., and Lee, J. D., "Airfoil Aerodynamics in Icing Conditions," Journal of Aircraft, Vol. 23, No. 1, January 1986.

23. Korkan, K. D.., Cross, E. J., Jr., and Cornell, C.C., "Experimental Aerodynamic Characteristics of an NACA 0012 Airfoil with Simulated Ice," Journal of Aircraft, Vol. 22, No. 2, February, 1985.

24. Flemming, R.J., and Lednicer, D. A., "Correlation of Icing Relationships with Airfoil and Rotorcraft Icing Data," Journal of Aircraft, Vol. 23, No. 10, October 1986.

25. Bragg, M. B., Khodadoust, A., Soltani, R., Wells, S. and Kerho, M., "Aerodynamic Measurements on a Finite Wing with Simulated Ice," AIAA-91-3217-CP, 1991.

26. Ranaudo, R.J., Batterson, J. G., Reehorst, A. L., Bond, T. H., and O'Mara, T. M., "Effects of Horizontal Tail Ice on Longitudinal Aerodynamic Derivatives," Journal of Aircraft, Vol. 28, No. 3, March 1991.

27. Potapczuk, M. G., "Numerical Analysis of an NACA 0012 Airfoil with Leading-Edge Ice Accretions," Journal of Aircraft, Vol. 25, No. 3, March 1988.

28. Cebeci, T., "Calculation of Flow over Iced Airfoils", AIAA Journal, Vol. 27, No. 7, July 1989.

29. Cebeci, T., Chen, H. H., and Alemdaroglu, N., "Fortified LEWICE with Viscous Effects," Journal of Aircraft, Vol. 28, No. 9, Sept. 1991.

30. Ashby, Dale L., Dudley, Michael R., and Iguchi, Steven K. "Development and Validation of an Advanced Low-Order Panel Method." NASA TM-101024, October 1988.

31. Ashby, Dale L., Dudley, Michael R., and Iguchi, Steven K. "Potential Flow Theory and Operation Guide for the Panel Code PMARC," NASA TM-102851, March 1990.

32. Pulliam, T. H., "Euler and Thin-Layer Navier-Stokes Codes: ARC2D, ARC3D," Notes for Computational Fluid Dynamics User's Workshop, The University of Tennessee Space Institute, TN, 1984.

33. Shaw, R.J., Potapczuk, M. G., and Bidwell, C. S., "Predictions of Airfoil Aerodynamic Performance Degradation Due to Icing," NASA TM 101434, Prepwered for the Fourth Symposium on Numerical and Physical Aspects of Aerodynamic Flows, January, 1989.

34. Maskew, Brian. "Program VSAERO Theory Document--a Computer Program for Calculating Nonlinear Aerodynamic Characteristics of Arbitrary Configurations." NASA CR-4023, September 1987.

35. Sorenson, R. L., "A Program to Generate Two-dimensional Grids About Airfoils and Other Shapes by the Use of Poison's Equations," NASA TM 81198, May 1980.

36. Nathman, J. K., and Strash, D. J., "Calculated Longitudinal Stability of an Aircraft With Ice Accumulation On Its Horizontal Stabilizer," Analytical Methods Report 8901, Analytical Methods, Inc., Redmond, WA, August, 1989.

37. Lan, C. E. "Theoretical Prediction of Wing Rocking." Paper No. 32 in AGARD CP-386, presented at the AGARD Symposium on Unsteady Aerodynamics--Fundamentals and Applications to Aircraft Dynamics, May 6-9,1985, Gottingen, FRG.

38. Abbott, I. H., and von Doenhoff, A. E., Theory of Wing Sections, Dover Publications, Inc., New York, 1959.

39. Tseng, J. B., and Lan, C. E. "Calculation of Aerodynamic Characteristics of Airplane Configurations at High Angles of Attack." NASA CR-4182, October 1988.

40. Lan, C. E., Emdad, H., Chin, S.; Sundaram, P., and Mehrotra, S. C., "Calculation of High Angle-of-Attack Aerodynamics of Fighter Configurations." AIAA Paper 89-2188CP, August 1989.

41. Degani, D. and Schiff, L. B., "Computation of Turbulent Supersonic Flows around Pointed Bodies having Crossflow Separation," Journal of Computational Physics, Vol. 66, 1986.

42. Walatka, P. P., Bunig, P. G. Pierce, L. and Elson, P. A., PLOT3D User's Manual, NASA TM 101067, July 1992.

43. Zaman, K. B. M. Q., and Potapczuk, M. G., "The Low Frequency Oscillation in the Flow Over a NACA0012 Airfoil with an 'Iced' Leading Edge", Low Reynolds Number Aerodynamics, T. J. Mueller (Editor), Springer-Verlag Berlin, Heidelberg, 1989.

44. Potapczuk, M. G. Navier-Stokes Analysis of Airfoils with Leading Edge Ice Accretions, PhD. Dissertation, University of Akron, 1989.

45. Bragg, M. B. and Khodadoust, A., "Effect of Simulated Glaze Ice On A Rectangular Wing," AIAA 86-0484, 24th Aerospace Sciences Meeting, Reno, NV, Jan. 1986.

46. Critzos, C. C., Heyson, H. H. and Boswinkle, R. W., Jr., "Aerodynamic Characteristics of NACA 0012 Airfoil Section at Angles of Attack From 0° to 180°", NACA TN 3361, January 1955.

47. Maarsingh, R. A., Labrujere, Th. E. and Smith, J., "Accuracy of Various Wall-Correction Methods for 3D Subsonic Wind-Tunnel

Testing", AGARD CP-429, Naples, Italy, 28 September-1 October 1987.

48. Loftin, L. K., Jr. and Smith, H. A., "Aerodynamic Characteristics of 15 NACA Airfoil Sections at Seven Reynolds Numbers From 0.7 x 10^6 to 9.0 x 10^6", NACA TN 1945, October 1949.

49. Personal Telephone Conversation with D. Ashby, NASA Ames Research Center, April 5, 1993.

50. Kohlman, David L., "Aircraft Icing: Meteorology, Protective Systems, Instrumentation and Certification," University of Kansas 1992 Aerospace Short Courses, October 13-15, 1992, Lawrence, KS.

Appendix A.

Makefile For Modified PMARC

This makefile contains a listing of all of the subroutines necessary to compile the Modified PMARC of this study. All of the subroutines listed except RESTPMARC.f and SURFINF.f have been modified or created to implement the changes of this study. All of these modified or created subroutines are included in the following appendices for complete documentation of this work.

```
mpmarc: dpmarc.o aerodat.o restpmarc.o search.o viscdata.o jets.o\
            wakinfl.o rhs.o idubpot.o surfinf.o length.o
        xlf -o mpmarc dpmarc.o aerodat.o restpmarc.o search.o\
                viscdata.o jets.o wakinfl.o rhs.o idubpot.o\
                surfinf.o length.o
dpmarc.o: dpmarc.f
        xlf -qdpc -O -c dpmarc.f
aerodat.o: aerodat.f
        xlf -qdpc -O -c aerodat.f
restpmarc.o: restpmarc.f
        xlf -qdpc -O -c restpmarc.f
search.o: search.f
        xlf -qdpc -O -c search.f
viscdata.o: viscdata.f
        xlf -qdpc -O -c viscdata.f
jets.o: jets.f
        xlf -qdpc -O -c jets.f
wakinfl.o: wakinfl.f
        xlf -qdpc -O -c wakinfl.f
rhs.o: rhs.f
        xlf -qdpc -O -c rhs.f
idubpot.o: idubpot.f
        xlf -qdpc -O -c idubpot.f
surfinf.o: surfinf.f
        xlf -qdpc -O -c surfinf.f
length.o: length.f
        xlf -qdpc -O -c length.f
```

Appendix B.

Main Program DPMARC.f

```
      PROGRAM PMARC
C
C-----------------------------------------------------------------------
C
C  MASTER VERSION NUMBER:      Release Version 12.02, 01/16/90
C
C  PURPOSE:    MAIN DRIVER FOR PMARC PROGRAM
C
C  CALLED BY:  NONE
C
C  EXTERNAL REFERENCES:   OPENF,JOBDATA,SURFGEN,SURFPAN,NABORS,SURFINF,WAKINIT,
C                         WAKPAN,WAKINFL,SOLVER,WAKDUB,NEUMANN,AERODAT,
C                         CORNERPT,WAKSTEP,STRMLIN,VSCAN,PATH
C
C  ENVIRONMENT:  VAX/VMS FORTRAN,CRAY CFT77 FORTRAN,
C                MACINTOSH DCM MACTRAN PLUS 3.0
C
C  AUTHOR:  Dale Ashby,
C           MS 247-2, NASA Ames Research Center, Moffett Field, CA.  94035
C
C  DEVELOPMENT HISTORY:
C       DATE   INITIALS   DESCRIPTION
C     1/16/90    DLA      CHANGED SOME LOGICAL EXPRESSIONS WHICH COMPARED
C                         A VARIABLE TO 0.0 IN IF STATEMENTS SO THAT THE
C                         VARIABLE (OR IT'S ABSOLUTE VALUE  WAS COMPARED
C                         TO EPS (0.000001).  THIS WAS DONE SEVERAL PLACES
C                         TO CORRECT PRECISION PROBLEMS ON THE MAC.
C
C-----------------------------------------------------------------------
C
C
C  CODE DIMENSIONING PARAMETERS
C
C  NUMBER OF SURFACE PANELS ALLOWED
C
      PARAMETER (NSPDIM = 4000)
C
C  NUMBER OF NEUMANN PANELS ALLOWED
C
      PARAMETER (NNPDIM = 50)
C
C  NUMBER OF PATCHES ALLOWED
C
      PARAMETER (NPDIM = 20)
C
C  NUMBER OF BASIC POINTS ALLOWED FOR SECTION DEFINITION
C  (ALSO NUMBER OF SECTIONS ALLOWED PER PATCH
C  (ALSO NUMBER OF ROWS OR COLUMNS + 1 ALLOWED ON A PATCH
C  CAUTION:  DO NOT SET THIS PARAMETER TO LESS THAN 50
C
      PARAMETER (NBPDIM = 100)
C
C  NUMBER OF WAKE PANELS ALLOWED
C
      PARAMETER (NWPDIM = 1500)
C
C  NUMBER OF WAKE COLUMNS ALLOWED ON EACH WAKE
C
      PARAMETER (NWCDIM = 50)
C
C  NUMBER OF WAKES ALLOWED
C
      PARAMETER (NWDIM = 50)
C
C  NUMBER OF SCAN VOLUMES OF EACH TYPE ALLOWED
C
      PARAMETER (NSVDIM = 10)
C
C  NUMBER OF POINTS PER OFF-BODY STREAMLINE ALLOWED
```

```
c
      PARAMETER (NSLPDIM = 1000)
c
c  NUMBER OF GROUPS OF PANELS ON WHICH NONZERO NORMAL VELOCITY IS PRESCRIBED
c
      PARAMETER (NVELDIM = 200)
c
c  NUMBER OF LINES AT A TIME TO BE READ IN FOR THE INFLUENCE COEF. MATRIX
c  IN THE SOLVER ROUTINE (BUFFERED INPUT FROM THE SCRATCH FILE)
c  (CAUTION:  DO NOT SET LARGER THAN ONE UNLESS YOU ARE SURE YOU HAVE
c  ENOUGH MEMORY TO HANDLE BUFFERED INPUT.)
c
      PARAMETER (MATBUF = 1)
c
c  NUMBER OF WAKE CORNER POINTS ALLOWED
c
      PARAMETER (NWCPDIM=(NWPDIM + 1)*2)
c
c  NUMBER OF SURFACE CORNER POINTS ALLOWED
c
      PARAMETER (NSCPDIM=(NSPDIM + 1)*2)
c
c  NUMBER OF EDGE PANELS ALLOWED ON A PATCH
c
      PARAMETER (NEPDIM = NBPDIM * 4)
c
ctnm  number of viscous data points to be read in 3/26/93
c
      parameter (nvpts = 30)
ctnm RLXFAC added 2/5/93
      COMMON/ CONST    / PI,      EPS,     FOURPI,  CBAR,
     +                   SSPAN,   SREF,    RMPX,   RMPY,     RMPZ,
     +                   MAXIT,    SOLRES,RLXFAC,
     +                   RCORS,   RFF
      COMMON/ INTERNAL / NCZONE,NCZPAN,CZDUB,VREF

ctnm added to iterate a solution with viscous data 1/15/93
ctnm updated to include section drag data 2/22/93
ctnm updated to include section moment and last pass info 3/19/93
ctnm updated to include section lift data 3/23/93
      common/ iterate /COLCLS(NPDIM,NBPDIM),COLCDS(NPDIM,NBPDIM),
     +                 COLCMS(NPDIM,NBPDIM),TCLS,TCDS,tcms,
     +                 dalpha(npdim,nbpdim),cd2dpt(npdim,nbpdim),
     +                 cm2dpt(npdim,nbpdim),last,cl2dpt(npdim,nbpdim)

      COMMON/ TSTEP     / NTSTPS, ITSTEP
      COMMON/ NEWNAB / KPAN(NSPDIM),KSIDE(NSPDIM),NEWNAB(NSPDIM),
     +                 NEWSID(NSPDIM),NBCHGE
      COMMON/ PNABOR / NABOR(4,NSPDIM),NABSID(4,NSPDIM)
      COMMON/ NUM      / NPAN,NPATCH,NWPAN,NWAKE,NCOMP,NASSEM
      COMMON/ CNSET    / ALPHA,ALDEG,YAW,YAWDEG,
     +                   BETA,WIND(3,3),PHIDOT,THEDOT,PSIDOT,
     +                   COMPOP,SYM,GPR,VINF,VSOUND
      COMMON/ PATNAM   / PNAME(6,NPDIM)
      COMMON/ PATCHES  / IDENT(NPDIM), IPAN(NPDIM), KLASS(NPDIM),
     +                   KOMP(NPDIM),   LPAN(NPDIM),  NCOL(NPDIM),
     +                   NPANS(NPDIM),  NROW(NPDIM)
      COMMON/ PRINT    / LSTINP,    LSTOUT, LSTGEO, LSTNAB, LSTWAK,
     +                   LSTFRQ, LSTCPV ,LSTHLD, LSTJET
      COMMON/ VELSET   / NOCF(NVELDIM),NOCL(NVELDIM),NORF(NVELDIM),
     +                   NORL(NVELDIM),NORPCH(NVELDIM),NORSET,
     +                   VNORM(NVELDIM),NVSJETIN(NPDIM)
      COMMON/ RUNCNTRL/ LENRUN
      COMMON/ SOLUTION  / SIG(NSPDIM), DUB(NSPDIM), PDUB(NSPDIM),
     +                    WDUB(NWPDIM), VX(NSPDIM), VY(NSPDIM),
     +                    VZ(NSPDIM), VXR(NNPDIM), VYR(NNPDIM),
     +                    VZR(NNPDIM),DIAG(NSPDIM),
     +                    RHSV(NSPDIM), VNORMAL(NSPDIM), CPDUB(NSCPDIM)
      COMMON/ SPANEL  / XC(NSPDIM), YC(NSPDIM), ZC(NSPDIM),
     +                  PCS(3,3,NSPDIM), AREA(NSPDIM), PFF(NSPDIM),
     +                  CPSX(NSCPDIM), CPSY(NSCPDIM), CPSZ(NSCPDIM),
     +                  ICPS(NPDIM), KPTYP(NSPDIM),SMP(NSPDIM),
     +                  SMQ(NSPDIM)
      COMMON/ SCRFILES / JSFIL(4)
      COMMON/ UNITS    / UNITS
      COMMON/ UNSTDY   / OMEGA(3,10), VFR(3,10), ITSTEP
```

```
ctnm  VISCOUS added for iteration with viscous data 1/19/93
ctnm  dimensions increased to nvpts 3/26/93

      COMMON/ VISCOUS / IVISCS,IDENTV(NPDIM),IVPRNT,NVISC,
     +                  NPVMAX(10),ALP2D(10,nvpts),CL2D(10,nvpts),
     +                  CD2D(10,nvpts),CM2D(10,nvpts),ALPZRO(10),
     +                  rhsinc(NSPDIM)

      COMMON/ WAKNAM/ WNAME(6,NWDIM)
      COMMON/ WAKES / NWCOL(NWDIM), NWROW(NWDIM), IWPAN(NWDIM),
     +                LWPAN(NWDIM), IDENTW(NWDIM),
     +                KWPU(NWCDIM,NWDIM), KWPL(NWCDIM,NWDIM),
     +                PHIU(NWCDIM,NWDIM), PHIL(NWCDIM,NWDIM),
     +                IFLEXW(NWDIM)
      COMMON/ WPANEL  / XCW(NWPDIM), YCW(NWPDIM), ZCW(NWPDIM),
     +                  PCSW(3,3,NWPDIM), AREAW(NWPDIM),
     +                  PFFW(NWPDIM),
     +                  CPWX(NWCPDIM), CPWY(NWCPDIM), CPWZ(NWCPDIM),
     +                  ICPW(NWDIM)
      dimension dubic(nspdim,matbuf)
      LOGICAL SYM,GPR
C
C
      CHARACTER*15 UNITS
      CHARACTER*4 PNAME
      CHARACTER*4 WNAME
      CALL OPENF
C
C  Rewind all scratch files from 17 to 20 and assign unit numbers
C
ctnm added 1/29/93 for wakinfl routine, turns off viscous output
      ivprnt=0

      DO 1 L=17,20
      REWIND L
      JSFIL(L-16) = L
    1 CONTINUE
C
C  READ IN BASIC DATA AND GEOMETRY INPUT
C
      CALL JOBDATA
      CALL SURFGEN
      IF(LENRUN.EQ.1)GO TO 50
C
C  FORM PANEL PARAMETERS AND PANEL NEIGHBORS
C
      CALL SURPAN
      CALL NABORS
      IF(LENRUN.EQ.2)GO TO 50
C
C  FORM SURFACE PANEL INFLUENCE COEFFICIENTS
C
      IF(LENRUN.NE.3)CALL SURFINF
C
C  READ IN WAKE INPUT INFORMATION AND FORM WAKE STARTING PARAMETERS
C
      CALL WAKINIT
      IF(LENRUN.EQ.3)THEN
        CALL WAKPAN
        GO TO 50
      ENDIF
C
C  START THE TIME STEP LOOP
C
      DO 10 ITSTEP=1,NTSTPS
      WRITE(16,61)ITSTEP
      TSTIME = ITSTEP * DTSTEP
C
C  SET PRINT CONTROL HOLD FOR THIS TIME STEP
C
      IF(ITSTEP.EQ.1.AND.LSTFRQ.NE.0)THEN
        LSTHLD = 0
        GO TO 30
      ENDIF
      IF(ITSTEP.EQ.NTSTPS)THEN
```

```
          LSTHLD = 0
          GO TO 30
        ENDIF
        ITEST = ITSTEP
   20   CONTINUE
        ITEST = ITEST - LSTFRQ
        IF(ITEST.GT.0)GO TO 20
        IF(ITEST.EQ.0)THEN
          LSTHLD = 0
        ELSE
          LSTHLD = 1
        ENDIF
   30   CONTINUE
        CALL PATH(TSTIME)
        CALL WAKPAN
C
C  FORM THE RIGHT HAND SIDE VECTOR
C
        CALL WAKINFL
C
C  SOLVE THE MATRIX EQUATION FOR THE UNKNOWN DOUBLET STRENGTHS
C
        CALL SOLVER
C
C  DISTRIBUTE THE PROPER DOUBLET STRENGTHS ON THE WAKE PANELS
C
        CALL WAKDUB
C
C  COMPUTE THE SURFACE VELOCITIES, PRESSURE COEFFICIENTS, AND MACH NUMBER
C  AT THE PANEL CENTROIDS AND CORNER POINTS, AND FORCE AND MOMENT COEFFICIENTS
C
        CALL NEUMANN
        CALL AERODAT

ctnm  if a viscous case, do the following

        if (IVISCS.eq.1)then
          ivprnt=1
          call viscdata
          ivprnt=0
        endif

        CALL CORNERPT
C
C  STEP THE WAKE
C
        IF(ITSTEP.NE.NTSTPS)THEN
          CALL WAKSTEP
        ENDIF
   10   CONTINUE
C
C  COMPUTE SURFACE STREAMLINES AND BOUNDARY LAYER CALCULATIONS IF REQUESTED
C
C        CALL STLINE
C
C  PERFORM OFFBODY VELOCITY SCAN AND STREAMLINE CALCULATIONS
C
        CALL VSCAN
        CALL STRMLIN
   50   CONTINUE

ctnm  close added to close all units (gets rid of scratch files) 1/8/93

        do 51 iunit=11,20
          close (iunit)
   51   continue
        STOP
   61   FORMAT(//1H1,'TIME STEP',I4)
        END
```

Appendix C.

Subroutine AERODAT.f

```
*DECK AERODAT
      SUBROUTINE AERODAT
C
C-----------------------------------------------------------------------
C
C  PURPOSE:     EVALUATES SURFACE VELOCITY VECTOR AND PRESSURE COEFFICIENT
C               AT EACH CONTROL POINT AND INTEGRATES PRESSURE COEFFICIENTS
C               TO GET FORCES AND MOMENTS
C
C  CALLED BY:  PMARC
C
C  EXTERNAL REFERENCES:   SCHEME
C
C  ENVIRONMENT:  VAX/VMS FORTRAN,CRAY CFT77 FORTRAN,
C                MACINTOSH DCM MACTRAN PLUS 3.0
C
C  AUTHOR:  Dale Ashby,
C           MS 247-2, NASA Ames Research Center, Moffett Field, CA.  94035
C
C  DEVELOPMENT HISTORY:
C       DATE   INITIALS  DESCRIPTION
C
C-----------------------------------------------------------------------
C
C
C  CODE DIMENSIONING PARAMETERS
C
C  NUMBER OF SURFACE PANELS ALLOWED
C
      PARAMETER (NSPDIM = 4000)
C
C  NUMBER OF NEUMANN PANELS ALLOWED
C
      PARAMETER (NNPDIM = 50)
C
C  NUMBER OF PATCHES ALLOWED
C
      PARAMETER (NPDIM = 20)
C
C  NUMBER OF BASIC POINTS ALLOWED FOR SECTION DEFINITION
C  (ALSO NUMBER OF SECTIONS ALLOWED PER PATCH)
C  (ALSO NUMBER OF ROWS OR COLUMNS + 1 ALLOWED ON A PATCH)
C  CAUTION:  DO NOT SET THIS PARAMETER TO LESS THAN 50)
C
      PARAMETER (NBPDIM = 100)
C
C  NUMBER OF WAKE PANELS ALLOWED
C
      PARAMETER (NWPDIM = 1500)
C
C  NUMBER OF WAKE COLUMNS ALLOWED ON EACH WAKE
C
      PARAMETER (NWCDIM = 50)
C
C  NUMBER OF WAKES ALLOWED
C
      PARAMETER (NWDIM = 50)
C
C  NUMBER OF SCAN VOLUMES OF EACH TYPE ALLOWED
C
      PARAMETER (NSVDIM = 10)
C
C  NUMBER OF POINTS PER OFF-BODY STREAMLINE ALLOWED
C
      PARAMETER (NSLPDIM = 1000)
C
C  NUMBER OF GROUPS OF PANELS ON WHICH NONZERO NORMAL VELOCITY IS PRESCRIBED
C
      PARAMETER (NVELDIM = 200)
C
```

```
C  NUMBER OF LINES AT A TIME TO BE READ IN FOR THE INFLUENCE COEF. MATRIX
C  IN THE SOLVER ROUTINE (BUFFERED INPUT FROM THE SCRATCH FILE)
C  (CAUTION:  DO NOT SET LARGER THAN ONE UNLESS YOU ARE SURE YOU HAVE
C  ENOUGH MEMORY TO HANDLE BUFFERED INPUT.)
C
      PARAMETER (MATBUF = 1)
C
C  NUMBER OF WAKE CORNER POINTS ALLOWED
C
      PARAMETER (NWCPDIM=(NWPDIM + 1)*2)
C
C  NUMBER OF SURFACE CORNER POINTS ALLOWED
C
      PARAMETER (NSCPDIM=(NSPDIM + 1)*2)
C
C  NUMBER OF EDGE PANELS ALLOWED ON A PATCH
C
      PARAMETER (NEPDIM = NBPDIM * 4)
C
ctnm  number of viscous data points to be read in 3/26/93
c
      parameter (nvpts = 30)
ctnm subscripts not needed due to streamlining of code 2/25/93
c     DIMENSION  PATFX(NPDIM),PATFY(NPDIM),PATFZ(NPDIM),
c    +   PATMX(NPDIM),PATMY(NPDIM),PATMZ(NPDIM),

      dimension
     +   SUMA(NPDIM),PCLW(NPDIM),PCDW(NPDIM),PCSW(NPDIM),
     +   PCMW(NPDIM),PCMYW(NPDIM),PCMRW(NPDIM),
     +   PCLB(NPDIM),PCDB(NPDIM),PCSB(NPDIM),PCMB(NPDIM),
     +   PCMYB(NPDIM),PCMRB(NPDIM),PCLS(NPDIM),PCDS(NPDIM),
     +   PCSS(NPDIM),PCMS(NPDIM),PCMYS(NPDIM),PCMRS(NPDIM),
     +   CCLW(10),CCDW(10),CCSW(10),CCMW(10),CCMYW(10),
     +   CCMRW(10),CCLB(10),CCDB(10),CCSB(10),
     +   CCMB(10),CCMYB(10),CCMRB(10),CCLS(10),CCDS(10),
     +   CCSS(10),CCMS(10),CCMYS(10),CCMRS(10),
     +   ACLW(10),ACDW(10),ACSW(10),ACMW(10),ACMYW(10),
     +   ACMRW(10),ACLB(10),ACDB(10),ACSB(10),
     +   ACMB(10),ACMYB(10),ACMRB(10),ACLS(10),ACDS(10),
     +   ACSS(10),ACMS(10),ACMYS(10),ACMRS(10),
     +   KLSS(NPDIM),
     +   N(4),NS(4),SXX(4),SYY(4),SZZ(4),ICPSSUB(4)
      COMMON/ PATNAM    / PNAME(6,NPDIM)
      COMMON/ PATCHES   / IDENT(NPDIM), IPAN(NPDIM), KLASS(NPDIM),
     +                    KOMP(NPDIM),  LPAN(NPDIM), NCOL(NPDIM),
     +                    NPANS(NPDIM), NROW(NPDIM)
      COMMON/ SPANEL  / XC(NSPDIM), YC(NSPDIM), ZC(NSPDIM),
     +                  PCS(3,3,NSPDIM), AREA(NSPDIM), PFF(NSPDIM),
     +                  CPSX(NSCPDIM), CPSY(NSCPDIM), CPSZ(NSCPDIM),
     +                  ICPS(NPDIM), KPTYP(NSPDIM),SMP(NSPDIM),
     +                  SMQ(NSPDIM)
      COMMON/ PRINT     / LSTINP,   LSTOUT, LSTGEO, LSTNAB, LSTWAK,
     +                    LSTFRQ, LSTCPV ,LSTHLD, LSTJET
      COMMON/ SOLUTION  / SIG(NSPDIM), DUB(NSPDIM), PDUB(NSPDIM),
     +                    WDUB(NWPDIM), VX(NSPDIM), VY(NSPDIM),
     +                    VZ(NSPDIM), VXR(NNPDIM), VYR(NNPDIM),
     +                    VZR(NNPDIM),DIAG(NSPDIM),
     +                    RHSV(NSPDIM), VNORMAL(NSPDIM), CPDUB(NSCPDIM)
ctnm RLXFAC added 2/5/93
      COMMON/ CONST     / PI,      EPS,     FOURPI,  CBAR,
     +                    SSPAN,  SREF,   RMPX,   RMPY,     RMPZ,
     +                    MAXIT,   SOLRES,RLXFAC,
     +                    RCORS,  RFF
      COMMON/ INTERNAL/ NCZONE,   NCZPAN,    CZDUB, VREF

ctnm added to iterate a solution with viscous data 1/15/93
ctnm updated to include section drag data 2/22/93
ctnm updated to include section moment and last pass info 3/19/93
ctnm updated to include section lift data 3/23/93
      common/ iterate /COLCLS(NPDIM,NBPDIM),COLCDS(NPDIM,NBPDIM),
     +                 COLCMS(NPDIM,NBPDIM),TCLS,TCDS,tcms,
     +                 dalpha(npdim,nbpdim),cd2dpt(npdim,nbpdim),
     +                 cm2dpt(npdim,nbpdim),last,cl2dpt(npdim,nbpdim)

      COMMON/ PNABOR / NABOR(4,NSPDIM),NABSID(4,NSPDIM)
      COMMON/ SCRFILES / JPLOT, JDUBIC,
```

```
     +                    JSORIC, IMC
      COMMON/ ONSET     / ALPHA,ALDEG,YAW,YAWDEG,
     +                    BETA,WIND(3,3),PHIDOT,THEDOT,PSIDOT,
     +                    COMPOP,SYM,GPR,VINF,VSOUND
      COMMON/ UNITS     / UNITS

ctnm  VISCOUS added for iteration with viscous data 1/19/93
ctnm  dimensions increased to 30 3/26/93
ctnm NOTE: maximum of 10 viscous data sets can be read in

      COMMON/ VISCOUS / IVISCS,IDENTV(NPDIM),IVPRNT,NVISC,
     +                  NPVMAX(10),ALP2D(10,nvpts),CL2D(10,nvpts),
     +                  CD2D(10,nvpts),CM2D(10,nvpts),ALPZRO(10),
     +                  rhsinc(NSPDIM)

      COMMON/ NUM       / NPAN,NPATCH,NWPAN,NWAKE,NCOMP,NASSEM
      COMMON/ TSTEP      / NTSTPS, ITSTEP
      COMMON/ UNSTDY    / OMEGA(3,10), VFR(3,10), DTSTEP
      LOGICAL SYM,GPR
      CHARACTER*15 UNITS
      CHARACTER*4 PNAME

ctnm declare a viscous wind transform matrix 2/11/93
      real vwind(3,3)
C
C  INITIALIZE VARIABLES
C
ctnm qscale is a scale factor to scale the dynamic pressure on the tail
c      this could be made part of the input for each patch
c      this routine currently assumes that the tail is patch #2 4/22/93

      qscale =0.9

      KMP = 1
      SUM = 0.0
      TCLW = 0.0
      TCDW = 0.0
      TCSW = 0.0
      TCMW = 0.0
      TCMYW = 0.0
      TCMRW = 0.0
      COLFX = 0.0
      COLFY = 0.0
      COLFZ = 0.0
      COLMX = 0.0
      COLMY = 0.0
      COLMZ = 0.0
      patfx = 0.
      patfy = 0.
      patfz = 0.
      patmx = 0.
      patmy = 0.
      patmz = 0.

ctnm subscripts dropped due to viscous corrections and
c          streamlining of code 2/25/93

      DO 1 I=1,NPATCH
c     PATFX(I) = 0.0
c     PATFY(I) = 0.0
c     PATFZ(I) = 0.0
c     PATMX(I) = 0.0
c     PATMY(I) = 0.0
c     PATMZ(I) = 0.0
      SUMA(I) = 0.0

ctnm initializations added due to code change 2/25/93

      pclw(i) = 0.
      pcdw(i) = 0.
      pcsw(i) = 0.
      pcmw(i) = 0.
      pcmyw(i) = 0.
      pcmrw(i) = 0.
      pclb(i) = 0.
      pcdb(i) = 0.
```

```
      pcsb(i) = 0.
      pcmb(i) = 0.
      pcmyb(i) = 0.
      pcmrb(i) = 0.
    1 CONTINUE
      DO 2 NK=1,NCOMP
      CCLW(NK) = 0.0
      CCDW(NK) = 0.0
      CCSW(NK) = 0.0
      CCMW(NK) = 0.0
      CCMYW(NK) = 0.0
      CCMRW(NK) = 0.0
    2 CONTINUE
      DO 3 NA=1,NASSEM
      ACLW(NA) = 0.0
      ACDW(NA) = 0.0
      ACSW(NA) = 0.0
      ACMW(NA) = 0.0
      ACMYW(NA) = 0.0
      ACMRW(NA) = 0.0
    3 CONTINUE

ctnm  added to initialize variables used with viscous data 1/18/93

      do 4 np=1,npatch
        do 5 nc=1,ncol(np)
          COLCLS(NP,NC)=0.
          COLCDS(NP,NC)=0.
          COLCMS(NP,NC)=0.
 5      continue
 4    continue

ctnm initialize the viscous wind axis transform matrix 2/11/93

      do 6 nx=1,3
        do 7 ny=1,3
          vwind(nx,ny)=0.
 7      continue
 6    continue

      CY = COS(YAW)
      SY = SIN(YAW)
      CAL = COS(ALPHA)
      SAL = SIN(ALPHA)
      K=0
      K34=0
      IF(VREF.LT.EPS)THEN
        RVINF = VINF
      ELSE
        RVINF = VREF
      ENDIF
C
C  SET SYMMETRY CONDITIONS
C
      IF(SYM)THEN
        RSYM = 0.5
        RSYM1 = 0.0
      ELSE
        RSYM = 1.0
        RSYM1 = 1.0
      ENDIF
C
C  FOR INTERNAL FLOWS, COMPUTE THE DOUBLET STRENGTH ON PANEL NCZPAN AS THE
C  AVERAGE OF THE DOUBLET STRENGTHS ON THE NEIGHBORING PANELS.
C
      IF(NCZONE.GT.0)THEN
        N1 = NABOR(1,NCZPAN)
        N2 = NABOR(2,NCZPAN)
        N3 = NABOR(3,NCZPAN)
        N4 = NABOR(4,NCZPAN)
        DENOM = 4.0
        IF(N1.LE.0)THEN
          DUBN1 = 0.0
          DENOM = DENOM - 1.0
        ELSE
          DUBN1 = DUB(N1)
```

```
        ENDIF
        IF(N2.LE.0)THEN
          DUBN2 = 0.0
          DENOM = DENOM - 1.0
        ELSE
          DUBN2 = DUB(N2)
        ENDIF
        IF(N3.LE.0)THEN
          DUBN3 = 0.0
          DENOM = DENOM - 1.0
        ELSE
          DUBN3 = DUB(N3)
        ENDIF
        IF(N4.LE.0)THEN
          DUBN4 = 0.0
          DENOM = DENOM - 1.0
        ELSE
          DUBN4 = DUB(N4)
        ENDIF
        IF(DENOM.LT.EPS)THEN
          DUB(NCZPAN) = CZDUB
        ELSE
          DUB(NCZPAN) = (DUBN1 + DUBN2 + DUBN3 + DUBN4)/DENOM
        ENDIF
      ENDIF
C
C  COMPUTE VELOCITIES AND CP AT SURFACE CONTROL POINTS
C
      K=0
      if(last.eq.1)then
         write(13,*) 'nccl    xle     xdist    zle     zdist   arearat
     +   cl2dpt    cm2dpt'
         write(13,*) '_____ _______ _______ - _______  _______ _________
     + __________  _________'
      endif
      DO 10 NP=1,NPATCH
      ID=IABS(IDENT(NP))
      KLSS(NP)=IABS(KLASS(NP))
      IF(LSTHLD.EQ.0)THEN
        WRITE(16,699)
        WRITE(16,600)NP,(PNAME(I,NP),I=1,6)
      ENDIF
      DO 20 NC=1,NCOL(NP)
      DSMAX = 0.0
      IF(LSTHLD.EQ.0)THEN
        WRITE(16,601)NC,UNITS,RVINF,UNITS
        WRITE(16,602)
      ENDIF
C
C  COMPUTE CIRCULATION FOR EACH COLUMN ON WING TYPE PATCHES
C
      IF(ID.EQ.1)THEN
        KTL = IPAN(NP) + (NC - 1) * NROW(NP)
        KTU = KTL + NROW(NP) - 1
        CIRC = (DUB(KTL) - DUB(KTU)) * FOURPI
      ENDIF
      DO 30 NR=1,NROW(NP)
      K=K+1
      IF(ID.EQ.3)K34=K34+1
C
C  SURFACE DOUBLET DIFFERENTIATION FOR COMPUTING VELOCITIES
C
      DELP = 0.0
      DELQ = 0.0
      DO 35 I=1,2
      II = I+2
      IFLAG = 0
      SS = 0.0
      NK = K
C
C  FIND NEIGHBORS FOR PANEL NK
C
      N(I) = NABOR(I,NK)
      N(II) = NABOR(II,NK)
      NS(I) = NABSID(I,NK)
      NS(II) = NABSID(II,NK)
```

```
      IF(NS(I).GT.0.AND.NS(II).GT.0)GOTO 36
      IF(NS(I).GT.0.OR.NS(II).GT.0)THEN
C
C  IF NEIGHBOR ON SIDE 1 OR 2 DOESN'T EXIST, FIND NEIGHBOR OF NEIGHBOR
C  ON SIDE 3 OR 4
C
        IF(NS(I).LE.0)THEN
          N(I) = NK
          NK = N(II)
          N(II) = NABOR(II,NK)
          NS(I) = NABSID(I,NK)
          NS(II) = NABSID(II,NK)
          IFLAG = II
          IF(NS(II).LE.0)THEN
            IFLAG = II+4
          ENDIF
C
C  IF NEIGHBOR ON SIDE 3 OR 4 DOESN'T EXIST, FIND NEIGHBOR OF NEIGHBOR
C  ON SIDE 1 OR 2
C
        ELSEIF(NS(II).LE.0)THEN
          N(II) = NK
          NK = N(I)
          N(I) = NABOR(I,NK)
          NS(II) = NABSID(II,NK)
          NS(I) = NABSID(I,NK)
          IFLAG = I
          IF(NS(I).LE.0)THEN
            IFLAG = I+4
          ENDIF
        ENDIF
      ELSE
C
C  IF NEIGHBORS DO NOT EXIST ON EITHER SIDE OF PANEL NK, WRITE MESSAGE
C  TO OUTPUT FILE AND GO TO NEXT PANEL
C
        WRITE(16,660)K,I,II
        DELP = 0.0
        DELQ = 0.0
        GO TO 35
      ENDIF
C
C  NOW THAT NEIGHBORS ARE IDENTIFIED, DO THE DIFFERENTIATION
C
   36 IF(I.EQ.1)THEN
        SK = SMQ(NK)
      ELSE
        SK = SMP(NK)
      ENDIF
      IF(IFLAG.LT.5)THEN
        IF(NS(I).EQ.1.OR.NS(I).EQ.3)THEN
          S1 = SMQ(N(I))
        ELSE
          S1 = SMP(N(I))
        ENDIF
        IF(NS(II).EQ.1.OR.NS(II).EQ.3)THEN
          S3 = SMQ(N(II))
        ELSE
          S3 = SMP(N(II))
        ENDIF
        SA = -(S1+SK)
        SB = (S3+SK)
        DB = (DUB(N(II))-DUB(NK))/SB
        DA = (DUB(N(I))-DUB(NK))/SA
        IF(IFLAG.EQ.I)THEN
          SS = SB
        ELSEIF(IFLAG.EQ.II)THEN
          SS = SA
        ENDIF
        IF(I.EQ.1)THEN
          DELQ = (DA*SB-DB*SA)/(SB-SA)+2*(DB-DA)/(SB-SA)*SS
        ELSE
          DELP = -((DA*SB-DB*SA)/(SB-SA)+2*(DB-DA)/(SB-SA)*SS)
        ENDIF
      ENDIF
C
```

```
C  IF ONLY ONE NEIGHBOR TO PANEL NK EXISTS, THEN JUST USE SIMPLE
C  DIFFERENCING BETWEEN PANELS TO GET THE DERIVATIVE
C
      IF(IFLAG.EQ.(I-4))THEN
        IF(NS(II).EQ.1.OR.NS(II).EQ.3)THEN
          S3 = SMQ(N(II))
        ELSE
          S3 = SMP(N(II))
        ENDIF
        SE = (S3+SK)
        IF(I.EQ.1)THEN
          DELQ = (DUB(N(II))-DUB(NK))/SE
        ELSE
          DELP = (DUB(N(II))-DUB(NK))/SE
        ENDIF
      ENDIF
      IF(IFLAG.EQ.(II-4))THEN
        IF(NS(I).EQ.2.OR.NS(I).EQ.4)THEN
          S1 = SMQ(N(I))
        ELSE
          S1 = SMP(N(I))
        ENDIF
        SA = (S1+SK)
        IF(I.EQ.1)THEN
          DELQ = (DUB(N(I))-DUB(NK))/SA
        ELSE
          DELP = (DUB(N(I))-DUB(NK))/SA
        ENDIF
      ENDIF
   35 CONTINUE
      CALL SCHEME(NROW(NP),NCOL(NP),IPAN(NP),K,ICPS(NP),
     +              ICPSSUB)
      DO 31 I=1,4
        SXX(I) = CPSX(ICPSSUB(I))
        SYY(I) = CPSY(ICPSSUB(I))
        SZZ(I) = CPSZ(ICPSSUB(I))
   31 CONTINUE
      EX3 = SXX(3) - XC(K)
      EX2 = SXX(2) - XC(K)
      EY3 = SYY(3) - YC(K)
      EY2 = SYY(2) - YC(K)
      EZ3 = SZZ(3) - ZC(K)
      EZ2 = SZZ(2) - ZC(K)
      XE3 = EX3 * PCS(1,1,K) + EY3 * PCS(2,1,K) + EZ3 * PCS(3,1,K)
      XE2 = EX2 * PCS(1,1,K) + EY2 * PCS(2,1,K) + EZ2 * PCS(3,1,K)
      YE3 = EX3 * PCS(1,2,K) + EY3 * PCS(2,2,K) + EZ3 * PCS(3,2,K)
      YE2 = EX2 * PCS(1,2,K) + EY2 * PCS(2,2,K) + EZ2 * PCS(3,2,K)
      TX = XE3 + XE2
      TY = YE3 + YE2
C
C  VELOCITY COMPUTED IN LOCAL PANEL COORDINATE SYSTEM
C
      VL = (TY * DELQ - SQRT(TX*TX + TY*TY) * DELP) * FOURPI/TX
      VM = -DELQ * FOURPI
      VN = SIG(K) * FOURPI
      DUBR = DUB(K) * FOURPI
C
C  TRANSFORM THE VELOCITY VECTOR TO GLOBAL COORDINATE SYSTEM
C
      VPX=(VL * PCS(1,1,K) + VM * PCS(1,2,K) + VN * PCS(1,3,K))
      VPY=(VL * PCS(2,1,K) + VM * PCS(2,2,K) + VN * PCS(2,3,K))
      VPZ=(VL * PCS(3,1,K) + VM * PCS(3,2,K) + VN * PCS(3,3,K))
      IF(ID.EQ.3)THEN
C
C  COMPUTE SURFACE VELOCITIES FOR NEUMANN PATCH
C
C  UPPER SURFACE VELOCITIES
C
        VMN=VNORMAL(K)-(VXR(K34)*PCS(1,3,K)+VYR(K34)*PCS(2,3,K)
     1            +VZR(K34)*PCS(3,3,K))
        VX(K) = VXR(K34) + VPX/2.0 + VMN * PCS(1,3,K)
        VY(K) = VYR(K34) + VPY/2.0 + VMN * PCS(2,3,K)
        VZ(K) = VZR(K34) + VPZ/2.0 + VMN * PCS(3,3,K)
C
C  LOWER SURFACE VELOCITIES
C
```

```
        VXL = VX(K) - VPX
        VYL = VY(K) - VPY
        VZL = VZ(K) - VPZ
        VMAGL = SQRT(VXL*VXL + VYL*VYL + VZL*VZL)
      ELSE
C
C  COMPUTE KINEMATIC VELOCITY VECTOR
C
        VSX = VFR(1,KMP) - OMEGA(2,KMP)*ZC(K) - OMEGA(3,KMP)*YC(K)
        VSY = VFR(2,KMP) + OMEGA(3,KMP)*XC(K) - OMEGA(1,KMP)*ZC(K)
        VSZ = VFR(3,KMP) + OMEGA(1,KMP)*YC(K) - OMEGA(2,KMP)*XC(K)
C
C  ADD KINEMATIC VELOCITY VECTOR
C
        VX(K) = VPX - VSX
        VY(K) = VPY - VSY
        VZ(K) = VPZ - VSZ
      ENDIF
C
C  COMPUTE PRESSURE COEFFICIENT AND LOCAL MACH NUMBER
C
        VXD = VX(K)
        VYD = VY(K)
        VZD = VZ(K)
        VMAG = SQRT(VXD**2 + VYD**2 + VZD**2)
        CP = 1 - (VMAG/RVINF)**2 + (2*FOURPI/(RVINF**2 * DTSTEP)) *
     +       (DUB(K) - PDUB(K))
        PMACH = VMAG/VSOUND
C
C  COMPUTE CP AND MACH NUMBER ON LOWER SURFACE OF NEUMANN PATCHES
C
      IF(ID.EQ.3)THEN
        CPL = 1 - (VMAGL/RVINF)**2 + (2*FOURPI/(RVINF**2 * DTSTEP)) *
     +       (DUB(K) - PDUB(K))
        PMACHL = VMAGL/VSOUND
      ENDIF
C
C  PERFORM PRANDTL-GLAUERT COMPRESSIBILITY CORRECTION
C
      IF(COMPOP.EQ.1)THEN
        CP = CP/BETA
        CPL = CPL/BETA
      ENDIF
C
C  COMPUTE THE FORCES AND MOMENTS ON EACH PANEL
C
      PFTOT = -CP * AREA(K)
      IF(ID.EQ.3)THEN
        PFTOT = -(CP - CPL) * AREA(K)
      ENDIF
      PFX = PFTOT * PCS(1,3,K)
      PFY = PFTOT * PCS(2,3,K)
      PFZ = PFTOT * PCS(3,3,K)
      PMX = PFZ * (YC(K) - RMPY) - PFY * (ZC(K) - RMPZ)
      PMY = PFX * (ZC(K) - RMPZ) - PFZ * (XC(K) - RMPX)
      PMZ = PFY * (XC(K) - RMPX) - PFX * (YC(K) - RMPY)
C
C  SUM UP FORCES AND MOMENTS FOR EACH COLUMN ON WING TYPE PATCHES
C
      IF(IDENT(NP).EQ.1)THEN
        COLFX = COLFX + PFX
        COLFY = COLFY + PFY
        COLFZ = COLFZ + PFZ
        COLMX = COLMX + PMX
        COLMY = COLMY + PMY
        COLMZ = COLMZ + PMZ
        SUM = SUM + AREA(K)
C
C  COMPUTE THE LEADING AND TRAILING EDGE COORDINATES FOR EACH COLUMN OF
C  PANELS ON TYPE 1 PATCHES.  ALSO COMPUTE THE CHORD LENGTH.
C
        IF(NR.EQ.1)THEN
          XTE = SXX(1) + SXX(4)
          YTE = SYY(1) + SYY(4)
          ZTE = SZZ(1) + SZZ(4)
        ENDIF
```

```
            DX = SXX(2) + SXX(3) - XTE
            DY = SYY(2) + SYY(3) - YTE
            DZ = SZZ(2) + SZZ(3) - ZTE
            DS = DX*DX + DY*DY + DZ*DZ
            IF(DS.GT.DSMAX)THEN
              DSMAX=DS
              XLE = (SXX(2) + SXX(3))/2.0
              YLE = (SYY(2) + SYY(3))/2.0
              ZLE = (SZZ(2) + SZZ(3))/2.0
            ENDIF
            DX = XTE/2.0 - XLE
            DY = YTE/2.0 - YLE
            DZ = ZTE/2.0 - ZLE
            CHORD = SQRT(DX*DX + DY*DY + DZ*DZ)
            YOVRSSPN = YLE/SSPAN
          ENDIF
C
C   SUM UP THE FORCES AND MOMENTS FOR EACH column 3/19/93
C
          PATFX = PATFX + PFX
          PATFY = PATFY + PFY
          PATFZ = PATFZ + PFZ
          PATMX = PATMX + PMX
          PATMY = PATMY + PMY
          PATMZ = PATMZ + PMZ
          SUMA(NP) = SUMA(NP) + AREA(K)/SREF
          IF(LSTHLD.EQ.0)THEN
            WRITE(16,603)K,XC(K),YC(K),ZC(K),DUBR,VXD,VYD,VZD,VMAG,CP,PMACH
            IF(ID.EQ.3)THEN
              WRITE(16,603)K,XC(K),YC(K),ZC(K),DUBR,VXL,VYL,VZL,VMAGL,CPL,
     +                    PMACHL
            ENDIF
          ENDIF
   30   CONTINUE
C
C   COMPUTE SECTION FORCE AND MOMENT COEFFICIENTS FOR WING TYPE PATCHES
C
ctnm added to allow modification to alpha for viscous calculations
c     alpha is modified for the dalpha calculated in rhs.f 2/11/93

          IF(IDENT(NP).EQ.1)THEN
          if(ivprnt.eq.0)then
            COLCLW = COLFX * WIND(1,3) + COLFY * WIND(2,3) +
     +      COLFZ * WIND(3,3)
            COLCDW = COLFX * WIND(1,1) + COLFY * WIND(2,1) +
     +      COLFZ * WIND(3,1)
            COLCSW = COLFX * WIND(1,2) + COLFY * WIND(2,2) +
     +      COLFZ * WIND(3,2)
            COLCMW = COLMX * WIND(1,2) + COLMY * WIND(2,2) +
     +      COLMZ * WIND(3,2)
            COLCMYW = COLMX * WIND(1,3) + COLMY * WIND(2,3) +
     +      COLMZ * WIND(3,3)
            COLCMRW = COLMX * WIND(1,1) + COLMY * WIND(2,1) +
     +      COLMZ * WIND(3,1)
            SUM = SUM/2.0
            COLCLW = COLCLW/SUM
            COLCDW = COLCDW/SUM
            COLCSW = COLCSW/SUM
            COLCMW = COLCMW/(SUM * CBAR)
            COLCMYW = COLCMYW/(SUM * SSPAN)
            COLCMRW = COLCMRW/(SUM * SSPAN)

          else
C
C   SET THE CURRENT ALPHA ANGLE AND THE CURRENT UNIT FREE-STREAM
c    VELOCITY VECTOR modified for viscous calculations 2/25/93
C
            VALPHA = ALPHA - dalpha(np,nc)
            vcal=cos(VALPHA)
            vsal=sin(VALPHA)
            VCA = CY * vcal
            VSA = CY * vsal
            UX = VCA
            UY = SY
            UZ = VSA
            US = SQRT(UX**2 + UY**2)
```

```
c
C  SET UP THE VWIND AXIS TRANSFORMATION MATRIX for this column
C   with viscous corrections 2/24/93

      VWIND(1,1) = UX
      VWIND(2,1) = UY
      VWIND(3,1) = UZ
      VWIND(1,2) = -UY/US
      VWIND(2,2) = UX/US
      VWIND(3,2) = 0.0
      VWIND(1,3) = -UX * UZ/US
      VWIND(2,3) = -UY * UZ/US
      VWIND(3,3) = US

C
C  COMPUTE SECTION FORCE AND MOMENT COEFFICIENTS FOR WING TYPE PATCHES
C      using the viscous wind matrix

       COLCLW = COLFX * VWIND(1,3) + COLFY * VWIND(2,3) +
     + COLFZ * VWIND(3,3)
       COLCDW = COLFX * VWIND(1,1) + COLFY * VWIND(2,1) +
     + COLFZ * VWIND(3,1)
       COLCSW = COLFX * VWIND(1,2) + COLFY * VWIND(2,2) +
     + COLFZ * VWIND(3,2)
       COLCMW = COLMX * VWIND(1,2) + COLMY * VWIND(2,2) +
     + COLMZ * VWIND(3,2)
       COLCMYW = COLMX * VWIND(1,3) + COLMY * VWIND(2,3) +
     + COLMZ * VWIND(3,3)
       COLCMRW = COLMX * VWIND(1,1) + COLMY * VWIND(2,1) +
     + COLMZ * VWIND(3,1)
       SUM = SUM/2.0
       COLCLW = COLCLW/SUM
       COLCDW = COLCDW/SUM
       COLCSW = COLCSW/SUM
       COLCMW = COLCMW/(SUM * CBAR)
       COLCMYW = COLCMYW/(SUM * SSPAN)
       COLCMRW = COLCMRW/(SUM * SSPAN)
C
       ENDIF
C
C  CONVERT SECTION COEFFICIENTS FROM WIND TO STABILITY AXES
C

ctnm subscripts added for viscous data calculations 1/15/93
       COLCLS(NP,NC) = COLCLW
ctnm  added to included viscous flowfield corrections to section drag 2/22/93
      if (ivprnt.eq.0)then
       COLCDS(NP,NC) = COLCDW * CY - COLCSW * SY
      else
       COLCDS(NP,NC) = COLCDW * CY - COLCSW * SY + cd2dpt(np,nc)
      endif

       COLCSS = COLCSW * CY + COLCDW * SY
       COLCMS(NP,NC) = COLCMW * CY - (SSPAN/CBAR) * COLCMRW * SY
       COLCMRS = COLCMRW * CY + (CBAR/SSPAN) * COLCMW * SY
       COLCMYS = COLCMYW
C
C  CONVERT SECTION COEFFICIENTS FROM WIND TO BODY AXES
C
ctnm added for viscous iterations 2/22/93

       if(ivprnt.eq.0)then

         COLCLB = COLCLW * CAL + COLCDW * CY * SAL + COLCSW * SY
     +   * SAL
         COLCDB = COLCDW * CY * CAL - COLCLW * SAL - COLCSW * SY
     +   * CAL
         COLCSB = COLCSW * CY + COLCDW * SY
         COLCMB = COLCMW * CY - (SSPAN/CBAR) * COLCMRW * SY
         COLCMRB = COLCMRW * CY * CAL + (CBAR/SSPAN) * COLCMW *
     +   SY * CAL - COLCMYW * SAL
         COLCMYB = COLCMYW * CAL + COLCMRW * CY * SAL + (CBAR/SSPAN) *
     +   COLCMW * SY * SAL

       else
```

```
            COLCLB = COLCLW * VCAL + COLCDW * CY * VSAL - COLCSW * SY
     +      * VSAL
            COLCDB = COLCDW * CY * VCAL - COLCLW * VSAL - COLCSW * SY
     +      * VCAL
            COLCSB = COLCSW * CY + COLCDW * SY
            COLCMB = COLCMW * CY - (SSPAN/CBAR) * COLCMRW * SY
            COLCMRB = COLCMRW * CY * VCAL + (CBAR/SSPAN) * COLCMW *
     +      SY * VCAL - COLCMYW * VSAL
            COLCMYB =COLCMYW*VCAL - COLCMRW * CY * VSAL - (CBAR/SSPAN) *
     +      COLCMW * SY * VSAL

          endif
ctnm reinitialization of SUM moved to scale 1-d drag increment
c         SUM = 0
          COLFX = 0
          COLFY = 0
          COLFZ = 0
          COLMX = 0
          COLMY = 0
          COLMZ = 0
C
C  WRITE SECTION PARAMETERS AND COEFFICIENTS TO OUTPUT FILE
C
ctnm if in a viscous iteration, skip these writes

         if (ivprnt.eq.1) goto 300
            WRITE(16,675)
            WRITE(16,676)XLE,YLE,ZLE,CHORD,CIRC,YOVRSSPN
            WRITE(16,677)
            WRITE(16,678)COLCLW,COLCDW,COLCSW,COLCMW,COLCMYW,COLCMRW

ctnm subscript added for iteration with viscous data 1/15/93
            WRITE(16,679)COLCLS(NP,NC),COLCDS(NP,NC),COLCSS,
     +                     COLCMS(NP,NC),COLCMYS,COLCMRS

            WRITE(16,680)COLCLB,COLCDB,COLCSB,COLCMB,COLCMYB,COLCMRB
 300  continue
      ENDIF

C
C  PUT THE PATCH FORCE AND MOMENT DATA IN WIND AXIS COEFFICIENT FORM
C

c        DO 40 NP=1,NPATCH
c           PCLW(NP) = PATFX(NP) * WIND(1,3) + PATFY(NP) * WIND(2,3) +
c    +      PATFZ(NP) * WIND(3,3)
c           PCDW(NP) = PATFX(NP) * WIND(1,1) + PATFY(NP) * WIND(2,1) +
c    +      PATFZ(NP) * WIND(3,1)
c           PCSW(NP) = PATFX(NP) * WIND(1,2) + PATFY(NP) * WIND(2,2) +
c    +      PATFZ(NP) * WIND(3,2)
c           PCMW(NP) = PATMX(NP) * WIND(1,2) + PATMY(NP) * WIND(2,2) +
c    +      PATMZ(NP) * WIND(3,2)
c           PCMYW(NP) = PATMX(NP) * WIND(1,3) + PATMY(NP) * WIND(2,3) +
c    +      PATMZ(NP) * WIND(3,3)
c           PCMRW(NP) = PATMX(NP) * WIND(1,1) + PATMY(NP) * WIND(2,1) +
c    +      PATMZ(NP) * WIND(3,1)
c           PCLW(NP) = PCLW(NP)/(SREF)
c           PCDW(NP) = PCDW(NP)/(SREF)
c           PCSW(NP) = PCSW(NP)/(SREF)
c           PCMW(NP) = PCMW(NP)/(SREF * CBAR)
c           PCMYW(NP) = PCMYW(NP)/(SREF * SSPAN)
c           PCMRW(NP) = PCMRW(NP)/(SREF * SSPAN)
c 40     CONTINUE
c
c  SUM UP COMPONENT, ASSEMBLY,AND TOTAL FORCE AND MOMENT
c  COEFFICIENTS IN WIND AXES
c
c      DO 50 NP=1,NPATCH
c      CCLW(KOMP(NP)) = CCLW(KOMP(NP)) + PCLW(NP)
c      CCDW(KOMP(NP)) = CCDW(KOMP(NP)) + PCDW(NP)
c      CCSW(KOMP(NP)) = CCSW(KOMP(NP)) + PCSW(NP)
c      CCMW(KOMP(NP)) = CCMW(KOMP(NP)) + PCMW(NP)
c      CCMYW(KOMP(NP)) = CCMYW(KOMP(NP)) + PCMYW(NP)
c      CCMRW(KOMP(NP)) = CCMRW(KOMP(NP)) + PCMRW(NP)
c      ACLW(KLASS(NP)) = ACLW(KLASS(NP)) + PCLW(NP)
```

```
c     ACDW(KLASS(NP)) = ACDW(KLASS(NP)) + PCDW(NP)
c     ACSW(KLASS(NP)) = ACSW(KLASS(NP)) + PCSW(NP)
c     ACMW(KLASS(NP)) = ACMW(KLASS(NP)) + PCMW(NP)
c     ACMYW(KLASS(NP)) = ACMYW(KLASS(NP)) + PCMYW(NP)
c     ACMRW(KLASS(NP)) = ACMRW(KLASS(NP)) + PCMRW(NP)
c     TCLW = TCLW + PCLW(NP)/RSYM
c     TCDW = TCDW + PCDW(NP)/RSYM
c     TCSW = TCSW + PCSW(NP) * RSYM1
c     TCMW = TCMW + PCMW(NP)/RSYM
c     TCMYW = TCMYW + PCMYW(NP) * RSYM1
c     TCMRW = TCMRW + PCMRW(NP) * RSYM1
c
c  CONVERT PATCH COEFFICIENTS FROM WIND TO STABILITY AXES
c
c     PCLS(NP) = PCLW(NP)
c     PCDS(NP) = PCDW(NP) * CY - PCSW(NP) * SY
c     PCSS(NP) = PCSW(NP) * CY + PCDW(NP) * SY
c     PCMS(NP) = PCMW(NP) * CY + (SSPAN/CBAR) * PCMRW(NP) * SY
c     PCMRS(NP) = PCMRW(NP) * CY - (CBAR/SSPAN) * PCMW(NP) * SY
c     PCMYS(NP) = PCMYW(NP)
c
c  CONVERT PATCH COEFFICIENTS FROM WIND TO BODY AXES
c
c     PCLB(NP) = PCLW(NP) * CAL + PCDW(NP) * CY * SAL - PCSW(NP) * SY
c    +     * SAL
c     PCDB(NP) = PCDW(NP) * CY * CAL - PCLW(NP) * SAL - PCSW(NP) * SY
c    +     * CAL
c     PCSB(NP) = PCSW(NP) * CY + PCDW(NP) * SY
c     PCMB(NP) = PCMW(NP) * CY - (SSPAN/CBAR) * PCMRW(NP) * SY
c     PCMRB(NP) = PCMRW(NP) * CY * CAL + (CBAR/SSPAN) * PCMW(NP) *
c    +     SY * CAL - PCMYW(NP) * SAL
c     PCMYB(NP) = PCMYW(NP) * CAL + PCMRW(NP) * CY * SAL + (CBAR/SSPAN)
c    +   * PCMW(NP) * SY * SAL
c  50 CONTINUE
c
c
c
c  PUT THE PATCH FORCE AND MOMENT DATA IN WIND AXIS COEFFICIENT FORM
c
          if(ivprnt.eq.0 .or. ident(np).ne.1)then
           PCLWW = PATFX * WIND(1,3) + PATFY *
     +       WIND(2,3) + PATFZ * WIND(3,3)
           PCDWW = PATFX * WIND(1,1) + PATFY *
     +       WIND(2,1) + PATFZ * WIND(3,1)
           PCSWW = PATFX * WIND(1,2) + PATFY *
     +       WIND(2,2) + PATFZ * WIND(3,2)
           PCMWW = PATMX * WIND(1,2) + PATMY *
     +       WIND(2,2) + PATMZ * WIND(3,2)
           PCMYWW = PATMX * WIND(1,3) + PATMY *
     +       WIND(2,3) + PATMZ * WIND(3,3)
           PCMRWW = PATMX * WIND(1,1) + PATMY *
     +       WIND(2,1) +  PATMZ * WIND(3,1)
           PCLWW = PCLWW/(SREF)
           PCDWW = PCDWW/(SREF)
           PCSWW = PCSWW/(SREF)
           PCMWW = PCMWW/(SREF * CBAR)
           PCMYWW = PCMYWW/(SREF * SSPAN)
           PCMRWW = PCMRWW/(SREF * SSPAN)

          else

           PCLWW = PATFX * vwind(1,3) + PATFY *
     +       vwind(2,3) + PATFZ * vwind(3,3)
           PCDWW = PATFX * vwind(1,1) + PATFY *
     +       vwind(2,1) + PATFZ * vwind(3,1)
           PCSWW = PATFX * vwind(1,2) + PATFY *
     +       vwind(2,2) + PATFZ * vwind(3,2)

ctnm check to see if this is the last pass 3/19/93
           if(last.eq.1)then
ctnm assume that the input 2-d data is referenced to the quarter
c    chord.  if it is not, this value will need to be added as
c    an input 3/19/19

              xmref = 0.25
ctnm define the perpendicular x,y and z distances for moments 3/18/93
```

```
ctnm only xdist is needed 3/23/93
                  xdist = rmpx - (xle + xmref * dx
                  ydist = yle + xmref * dy
                  zdist = zle + xmref * dz


        PATMX = PatFZ * (ydist - RMPY) - PatFY * (zdist - RMPZ
        PatMY = PatFX * (zdist - RMPZ) - PatFZ * (xdist - RMPX)
        PatMZ = PatFY * (Xdist - RMPX) - PatFX * (ydist - RMPY
ctnm  calculate the moment based on the x and z force componets 3/18/93
              endif
                  PCMWW = PATMX * vwind(1,2) + PATMY *
     +               vwind(2,2) + PATMZ * vwind(3,2)

              PCMYWW = PATMX * vwind(1,3) + PATMY *
     +          vwind(2,3) + PATMZ * vwind(3,3)
              PCMRWW = PATMX * vwind(1,1) + PATMY *
     +          vwind(2,1) +  PATMZ * vwind(3,1)

ctnm qscale is a scale factor to scale the dynamic pressure on the tail
c      this routine currently assumes that the tail is patch #2 4/22/93
            if (np.eq.2)then
              PCLWW = PCLWW*qscale/(SREF)
            else
              PCLWW = PCLWW/(SREF)
            endif
              PCSWW = PCSWW/(SREF)

              if(last.eq.0)then
                  PCDWW = PCDWW/(SREF)
                  PCMWW = PCMWW/(SREF * CBAR)
              else
ctnm define ratio of areas for lift, drag and moment corrections
ctnm   3/18/93

                  arearat = sum/sref

ctnm add the viscous drag component to PMARC's induced drag 3/18/93

                  PCDWW = PCDWW/(SREF) + cd2dpt(np,nc)* arearat

ctnm modified pitching moment for viscous moment and moments caused
ctnm      by friction drag 3/18/93
ctnm qscale is a scale factor to scale the dynamic pressure on the tail
c      this routine currently assumes that the tail is patch #2 4/22/93
                  if(np.eq.2)then
                     PCMWW = (qscale*cl2dpt(np,nc)*xdist -
     +               cm2dpt(np,nc) * chord + cd2dpt(np,nc)*zdist) *
     +               arearat/cbar
                  else
                     PCMWW = (cl2dpt(np,nc)*xdist + cm2dpt(np,nc) * chord +
     +                         cd2dpt(np,nc)*zdist) * arearat/cbar
                  endif
                  write(13,689)nc,xle,xdist,zle,zdist,arearat,
     +               cl2dpt(np,nc),cm2dpt(np,nc)
              endif

              PCMYWW = PCMYWW/(SREF * SSPAN)
              PCMRWW = PCMRWW/(SREF * SSPAN)
             endif

ctnm re-initialized due to use as a "working" variable
             sum = 0.
             patfx = 0.
             patfy = 0.
             patfz = 0.
             patmx = 0.
             patmy = 0.
             patmz = 0.
C
C  CONVERT PATCH COEFFICIENTS FROM WIND TO BODY AXES
C
ctnm  summation added due to viscous corrections 2/25/93
        if((ivprnt.eq.0 .or. ident(np).ne.1) then

          PCLB(np) = PCLB(np) + PCLWW * CAL + PCDWW * CY * SAL +
```

```
     +      PCSWW * SY * SAL
          PCDB(np) = PCDB(np) + PCDWW * CY * CAL - PCLWW * SAL -
     +      PCSWW * SY * CAL
          PCSB(np) = PCSB(np) + PCSWW * CY - PCDWW * SY
          PCMB(np) = PCMB(np) + PCMWW * CY - (SSPAN/CBAR) *
     +      PCMRWW * SY
          PCMRB(np) = PCMRB(np) + PCMRWW * CY * CAL + (CBAR/SSPAN) *
     +      PCMWW * SY * CAL - PCMYWW * SAL
          PCMYB(np) = PCMYB(np) - PCMYWW * CAL + PCMRWW * CY * SAL +
     +      (CBAR/SSPAN * PCMWW * SY * SAL

         else

          PCLB(np) = PCLB(np) + PCLWW * vcal + PCDWW *
     +      CY * vsal - PCSWW * SY * vsal
          PCDB(np) = PCDB(np) + PCDWW * CY * vcal - PCLWW *
     +      vsal - PCSWW * SY * vcal
          PCSB(np) = PCSB(np) + PCSWW * CY - PCDWW * SY
          PCMB(np) = PCMB(np) + PCMWW * CY - (SSPAN/CBAR) * PCMRWW *
     +     SY
          PCMRB(np) = PCMRB(np) + PCMRWW * CY * vcal + (CBAR/SSPAN) *
     +     PCMWW * SY * vcal - PCMYWW * vsal
          PCMYB(np) = PCMYB(np) + PCMYWW * vcal + PCMRWW * CY *
     +     vsal + (CBAR/SSPAN) * PCMWW * SY * vsal
         endif

         pclw(np) = pclw(np) + PCLWW
         pcdw(np) = pcdw(np) + PCDWW
         pcsw(np) = pcsw(np) + PCSWW
         pcmw(np) = pcmw(np) + PCMWW
         pcmyw(np) = pcmyw(np) + PCMYWW
         pcmrw(np) = pcmrw(np) + PCMRWW
 20      continue
 10      continue

C  SUM UP COMPONENT, ASSEMBLY,AND TOTAL FORCE AND MOMENT
C  COEFFICIENTS IN WIND AXES
C
        DO 50 NP = 1,NPATCH
          CCLW(KOMP(np)) = CCLW(KOMP(np)) + PCLW(np)
          CCDW(KOMP(np)) = CCDW(KOMP(np)) + PCDW(np)
          CCSW(KOMP(np)) = CCSW(KOMP(np)) + PCSW(np)
          CCMW(KOMP(np)) = CCMW(KOMP(np)) + PCMW(np)
          CCMYW(KOMP(np)) = CCMYW(KOMP(np)) + PCMYW(np)
          CCMRW(KOMP(np)) = CCMRW(KOMP(np)) + PCMRW(np)
          ACLW(KLASS(np)) = ACLW(KLASS(np)) + PCLW(np)
          ACDW(KLASS(np)) = ACDW(KLASS(np)) + PCDW(np)
          ACSW(KLASS(np)) = ACSW(KLASS(np)) + PCSW(np)
          ACMW(KLASS(np)) = ACMW(KLASS(np)) + PCMW(np)
          ACMYW(KLASS(np)) = ACMYW(KLASS(np)) + PCMYW(np)
          ACMRW(KLASS(np)) = ACMRW(KLASS(np)) + PCMRW(np)
          TCLW = TCLW + PCLW(np)/RSYM
          TCDW = TCDW + PCDW(np)/RSYM
          TCSW = TCSW + PCSW(np) * RSYM1
          TCMW = TCMW + PCMW(np)/RSYM
          TCMYW = TCMYW + PCMYW(np) * RSYM1
          TCMRW = TCMRW + PCMRW(np) * RSYM1
C
C  CONVERT PATCH COEFFICIENTS FROM WIND TO STABILITY AXES
C
          PCLS(np) = PCLW(np)
          PCDS(np) = PCDW(np) * CY - PCSW(np) * SY
          PCSS(np) = PCSW(np) * CY + PCDW(np) * SY
          PCMS(np) = PCMW(np) * CY + (SSPAN/CBAR) *
     +          PCMRW(np) * SY
          PCMRS(np) = PCMRW(np) * CY + (CBAR/SSPAN) *
     +          PCMW(np) * SY
          PCMYS(np) = PCMYW(np)
   50 CONTINUE
C
C  CONVERT COMPONENT COEFFICIENTS FROM WIND TO STABILITY AXIS
C
        DO 60 NK=1,NCOMP
        CCLS(NK) = CCLW(NK)
        CCDS(NK) = CCDW(NK) * CY - CCSW(NK) * SY
        CCSS(NK) = CCSW(NK) * CY + CCDW(NK) * SY
```

```
      CCMS(NK) = CCMW(NK) * CY + (SSPAN/CBAR) * CCMRW(NK) * SY
      CCMRS(NK) = CCMRW(NK) * CY + (CBAR/SSPAN) * CCMW(NK) * SY
      CCMYS(NK) = CCMYW(NK)
C
C  CONVERT COMPONENT COEFFICIENTS FROM WIND TO BODY AXES
C
      CCLB(NK) = CCLW(NK) * CAL + CCDW(NK) * CY * SAL + CCSW(NK) * SY
     +    * SAL
      CCDB(NK) = CCDW(NK) * CY * CAL - CCLW(NK) * SAL - CCSW(NK) * SY
     +    * CAL
      CCSB(NK) = CCSW(NK) * CY + CCDW(NK) * SY
      CCMB(NK) = CCMW(NK) * CY - (SSPAN/CBAR) * CCMRW(NK) * SY
      CCMRB(NK) = CCMRW(NK) * CY * CAL + (CBAR/SSPAN) * CCMW(NK) *
     +    SY * CAL - CCMYW(NK) * SAL
      CCMYB(NK) = CCMYW(NK) * CAL + CCMRW(NK) * CY * SAL + (CBAR/SSPAN)
     +   * CCMW(NK) * SY * SAL
   60 CONTINUE
C
C  CONVERT ASSEMBLY COEFFICIENTS FROM WIND TO STABILITY AXES
C
      DO 70 NA=1,NASSEM
      ACLS(NA) = ACLW(NA)
      ACDS(NA) = ACDW(NA) * CY - ACSW(NA) * SY
      ACSS(NA) = ACSW(NA) * CY + ACDW(NA) * SY
      ACMS(NA) = ACMW(NA) * CY + (SSPAN/CBAR) * ACMRW(NA) * SY
      ACMRS(NA) = ACMRW(NA) * CY + (CBAR/SSPAN) * ACMW(NA) * SY
      ACMYS(NA) = ACMYW(NA)
C
C  CONVERT ASSEMBLY COEFFICIENTS FROM WIND TO BODY AXES
C
      ACLB(NA) = ACLW(NA) * CAL + ACDW(NA) * CY * SAL + ACSW(NA) * SY
     +    * SAL
      ACDB(NA) = ACDW(NA) * CY * CAL - ACLW(NA) * SAL - ACSW(NA) * SY
     +    * CAL
      ACSB(NA) = ACSW(NA) * CY + ACDW(NA) * SY
      ACMB(NA) = ACMW(NA) * CY - (SSPAN/CBAR) * ACMRW(NA) * SY
      ACMRB(NA) = ACMRW(NA) * CY * CAL + (CBAR/SSPAN) * ACMW(NA) *
     +    SY * CAL - ACMYW(NA) * SAL
      ACMYB(NA) = ACMYW(NA) * CAL + ACMRW(NA) * CY * SAL + (CBAR/SSPAN)
     +   * ACMW(NA) * SY * SAL
   70 CONTINUE
C
C  CONVERT TOTAL COEFFICIENTS FROM WIND TO STABILITY AXES
C
      TCLS = TCLW
      TCDS = TCDW * CY - TCSW * SY
      TCSS = TCSW * CY + TCDW * SY
      TCMS = TCMW * CY + (SSPAN/CBAR) * TCMRW * SY
      TCMRS = TCMRW * CY + (CBAR/SSPAN) * TCMW * SY
      TCMYS = TCMYW
C
C  CONVERT TOTAL COEFFICIENTS FROM WIND TO BODY AXES
C
      TCLB = TCLW * CAL + TCDW * CY * SAL + TCSW * SY
     +    * SAL
      TCDB = TCDW * CY * CAL - TCLW * SAL - TCSW * SY
     +    * CAL
      TCSB = TCSW * CY + TCDW * SY
      TCMB = TCMW * CY - (SSPAN/CBAR) * TCMRW * SY
      TCMRB = TCMRW * CY * CAL + (CBAR/SSPAN) * TCMW *
     +    SY * CAL - TCMYW * SAL
      TCMYB = TCMYW * CAL + TCMRW * CY * SAL + (CBAR/SSPAN) *
     +    TCMW * SY * SAL
C
C  PRINT OUT ALL FORCE AND MOMENT DATA
C
ctnm if in a viscous iteration, skip these writes

        if (ivprnt.eq.1) goto 301
         WRITE(16,699)
         WRITE(16,604)
         WRITE(16,650)
C
C  WIND AXES COEFFICIENTS
C
         WRITE(16,607)
```

```
      WRITE(16,605)
      WRITE(16,606)
      DO 80 NP=1,NPATCH
      WRITE(16,620)NP,(PNAME(I,NP),I=1,6),PCLW(NP),PCDW(NP),PCSW(NP),
     +            PCMW(NP),PCMYW(NP),PCMRW(NP),SUMA(NP)
   80 CONTINUE
      WRITE(16,610)
      WRITE(16,625)
      DO 86 NK=1,NCOMP
      WRITE(16,622)NK,CCLW(NK),CCDW(NK),CCSW(NK),CCMW(NK),
     +            CCMYW(NK),CCMRW(NK)
   86 CONTINUE
      WRITE(16,612)
      WRITE(16,626)
      DO 92 NA=1,NASSEM
      WRITE(16,622)NA,ACLW(NA),ACDW(NA),ACSW(NA),ACMW(NA),
     +            ACMYW(NA),ACMRW(NA)
   92 CONTINUE
      WRITE(16,615)
      WRITE(16,627)
      WRITE(16,624)TCLW,TCDW,TCSW,TCMW,TCMYW,TCMRW
C
C  STABILITY AXES COEFFICIENTS
C
      WRITE(16,699)
      WRITE(16,608)
      WRITE(16,605)
      WRITE(16,606)
      DO 82 NP=1,NPATCH
      WRITE(16,620)NP,(PNAME(I,NP),I=1,6),PCLS(NP),PCDS(NP),PCSS(NP),
     +            PCMS(NP),PCMYS(NP),PCMRS(NP),SUMA(NP)
c     WRITE(13,620)NP,(PNAME(I,NP),I=1,6),PCLS(NP)
   82 CONTINUE
      WRITE(16,610)
      WRITE(16,625)
      DO 88 NK=1,NCOMP
      WRITE(16,622)NK,CCLS(NK),CCDS(NK),CCSS(NK),CCMS(NK),
     +            CCMYS(NK),CCMRS(NK)
   88 CONTINUE
      WRITE(16,612)
      WRITE(16,626)
      DO 94 NA=1,NASSEM
      WRITE(16,622)NA,ACLS(NA),ACDS(NA),ACSS(NA),ACMS(NA),
     +            ACMYS(NA),ACMRS(NA)
   94 CONTINUE
      WRITE(16,615)
      WRITE(16,627)
      WRITE(16,624)TCLS,TCDS,TCSS,TCMS,TCMYS,TCMRS
C
C  BODY AXES COEFFICIENTS
C
      WRITE(16,699)
      WRITE(16,609)
      WRITE(16,605)
      WRITE(16,628)
      DO 84 NP=1,NPATCH
      WRITE(16,620)NP,(PNAME(I,NP),I=1,6),PCLB(NP),PCDB(NP),PCSB(NP),
     +            PCMB(NP),PCMYB(NP),PCMRB(NP),SUMA(NP)
   84 CONTINUE
      WRITE(16,610)
      WRITE(16,629)
      DO 90 NK=1,NCOMP
      WRITE(16,622)NK,CCLB(NK),CCDB(NK),CCSB(NK),CCMB(NK),
     +            CCMYB(NK),CCMRB(NK)
   90 CONTINUE
      WRITE(16,612)
      WRITE(16,630)
      DO 96 NA=1,NASSEM
      WRITE(16,622)NA,ACLB(NA),ACDB(NA),ACSB(NA),ACMB(NA),
     +            ACMYB(NA),ACMRB(NA)
   96 CONTINUE
      WRITE(16,615)
      WRITE(16,631)
      WRITE(16,624)TCLB,TCDB,TCSB,TCMB,TCMYB,TCMRB
 301  continue
ctrm  output for data comparison, added 22 dec 92, if added 1/29/93
```

```
      if(ivprnt.eq.1.or.ivprnt.eq.0)then
        WRITE(13,624)REAL(ITSTEP),ALDEG,TCLS,TCDS,TCMS
      endif
      RETURN
C
C  FORMAT STATEMENTS
C
  600 FORMAT(/1X,'AERODYNAMIC DATA FOR PATCH',I5,11X,6A4/)
  601 FORMAT(/1X,'COLUMN',I5,40X,'VELOCITIES IN UNITS OF:  ',A15,
     +2X,'VINF = ',F10.4,A15/)
  602 FORMAT(1X,'PANEL',11X,'X',9X,'Y',9X,'Z',11X,'DUB',12X,'VX',10X,
     +'VY',10X,'VZ',11X,'V',10X,'CP',8X,'MACH'/)
  603 FORMAT(1X,I5,2X,3F10.3,2X,F12.4,2X,4F11.4,2X,F10.4,2X,F10.4)
  604 FORMAT(//30X,'*****************************'/30X,
     +'FORCE AND MOMENT COEFFICIENTS'/30X,'**********************',
     +'*******'//)
  605 FORMAT(30X,'PATCH COEFFICIENTS'/30X,'------------------'/)
  606 FORMAT(1X,'PATCH',10X,'NAME',18X,'CL',8X,'CD',8X,'CY',7X,'C_m',
     +7X,'C_n',7X,'C_l',5X,'PATCH AREA/SREF'/)
  607 FORMAT(/30X,'**********'/30X,'WIND AXES'/30X,'**********'/)
  608 FORMAT(/30X,'***************'/30X,'STABILITY AXES'/30X,
     +'***************'/)
  609 FORMAT(/30X,'**********'/30X,'BODY AXES'/30X,'**********'/)
  610 FORMAT(/30X,'COMPONENT COEFFICIENTS'/30X,'------------------',
     +'----'/)
  612 FORMAT(/30X,'ASSEMBLY COEFFICIENTS'/30X,'---------------------'/)
  615 FORMAT(/30X,'TOTAL COEFFICIENTS'/30X,'------------------'/)
  620 FORMAT(1X,I5,6A4,6F10.4,10X,F10.4)
  622 FORMAT(1X,I5,24X,6F10.4)
  624 FORMAT(30X,6F10.4)
  625 FORMAT(1X,'COMP',11X,'NAME',18X,'CL',8X,'CD',8X,'CY',7X,'C_m',
     +7X,'C_n',7X,'C_l'/)
  626 FORMAT(1X,'ASSEM',10X,'NAME',18X,'CL',8X,'CD',8X,'CY',7X,'C_m',
     +7X,'C_n',7X,'C_l'/)
  627 FORMAT(38X,'CL',8X,'CD',8X,'CY',7X,'C_m',
     +7X,'C_n',7X,'C_l'/)
  628 FORMAT(1X,'PATCH',10X,'NAME',18X,'CN',8X,'CA',8X,'CY',7X,'C_m',
     +7X,'C_n',7X,'C_l',5X,'PATCH AREA/SREF'/)
  629 FORMAT(1X,'COMP',11X,'NAME',18X,'CN',8X,'CA',8X,'CY',7X,'C_m',
     +7X,'C_n',7X,'C_l'/)
  630 FORMAT(1X,'ASSEM',10X,'NAME',18X,'CN',8X,'CA',8X,'CY',7X,'C_m',
     +7X,'C_n',7X,'C_l'/)
  631 FORMAT(38X,'CN',8X,'CA',8X,'CY',7X,'C_m',
     +7X,'C_n',7X,'C_l'/)
  650 FORMAT(1X,'NOTE:  If the geometry is panelled using a plane of',
     +' symmetry about the Y=0.0 plane, only the total force and'/1X,
     +'moment coefficients',
     +' will include the contribution from the image panels.'//)
  660 FORMAT(//1X,'PANEL',1X,I5,1X,'HAS NO NEIGHBORS ON SIDES',
     + 1X,I2,1X,'AND',1X,I2,1X,'.  THEREFORE TANGENTIAL SURFACE',
     +' VELOCITIES CANNOT BE COMPUTED FOR THIS PANEL.'//)
  675 FORMAT(/1X,'SECTION PARAMETERS'/8X,'XLE',7X,'YLE',7X,'ZLE',5X,
     +'CHORD',6X,'CIRC',5X,'YLE/SSPAN')
  676 FORMAT(1X,5F10.3,4X,F10.3)
  677 FORMAT(/18X,'CL',8X,'CD',8X,'CY',7X,'C_m',7X,'C_n',7X,'C_l'/)
  678 FORMAT(1X,'WIND',5X,6F10.4)
  679 FORMAT(1X,'STABILITY',6F10.4)
  680 FORMAT(1X,'BODY',5X,6F10.4)
  689 format(1x,i3,7(1x,f8.4))
  699 FORMAT(1H1)
      END
```

Appendix D.

Subroutine IDUBPOT.f

```
*DECK IDUBPOT
      SUBROUTINE IDUBPOT(scale,CXP,CYP,CZP,PJX,PJY,PJZ,
     +                   CNX,CNY,CNZ,XP,YP,ZP,CJK,CJK1)

ctnm  scale added to argument list to receive the scaling factor for the
c      doublet strength on the first wake 4/15/93
c      cjk1 brings back to wakinfl the value of cjk for the 1st wake column
C
C-----------------------------------------------------------------------------
C
C  PURPOSE:   THIS ROUTINE COMPUTES THE VELOCITY POTENTIAL INFLUENCE COEFFICIENT
C             ON PANEL K DUE TO A UNIT DISTRIBUTED DOUBLET ON PANEL J
C
C  CALLED BY:  WAKINFL
C
C  EXTERNAL REFERENCES:   CROSS
C
C  ENVIRONMENT:  VAX/VMS FORTRAN,CRAY CFT77 FORTRAN,
C                MACINTOSH DCM MACTRAN PLUS 3.0
C
C  AUTHOR:  Dale Ashby,
C          MS 247-2, NASA Ames Research Center, Moffett Field, CA.  94035
C
C  DEVELOPMENT HISTORY:
C       DATE   INITIALS  DESCRIPTION
C
C-----------------------------------------------------------------------------
C
C
ctnm CONST added to make EPS available to IDUBPOT 4/7/93
ctnm RLXFAC added 2/5/93
      COMMON/ CONST    / PI,      EPS,     FOURPI,  CBAR,
     +                   SSPAN,   SREF,    RMPX,  RMPY,     RMPZ,
     +                   MAXIT,   SOLRES,RLXFAC,
     +                   RCORS,   RFF
      REAL CXP(5),CYP(5),CZP(5)
      CXP(5)=CXP(1)
      CYP(5)=CYP(1)
      CZP(5)=CZP(1)
      PJKX = XP - PJX
      PJKY = YP - PJY
      PJKZ = ZP - PJZ
      PNJK=PJKX*CNX+PJKY*CNY+PJKZ*CNZ
      TMX=(CXP(3)+CXP(4))/2. - PJX
      TMY=(CYP(3)+CYP(4))/2. - PJY
      TMZ=(CZP(3)+CZP(4))/2. - PJZ
      TMS=SQRT(TMX*TMX+TMY*TMY+TMZ*TMZ)
C ***** CMX,CMY,CMZ COMPONENTS OF THE 'M' VECTOR                       *****
      CMX=TMX/TMS
      CMY=TMY/TMS
      CMZ=TMZ/TMS
C ***** CLX,CLY,CLZ COMPONENTS OF THE 'L' VECTOR                       *****
      CALL CROSS(CMX,CMY,CMZ,CNX,CNY,CNZ,CLX,CLY,CLZ)
      CJK = 0.
      CJK1 = 0.
        DO 10 NS=1,4
          AX = XP - CXP(NS)
          AY = YP - CYP(NS)
          AZ = ZP - CZP(NS)
          BX = XP - CXP(NS+1)
          BY = YP - CYP(NS+1)
          BZ = ZP - CZP(NS+1)
          A=SQRT(AX*AX + AY*AY + AZ*AZ)
          B=SQRT(BX*BX + BY*BY + BZ*BZ)
          XS = CXP(NS+1) - CXP(NS)
          YS = CYP(NS+1) - CYP(NS)
          ZS = CZP(NS+1) - CZP(NS)
          S=SQRT(XS*XS + YS*YS + ZS*ZS)
          SM=XS*CMX+YS*CMY+ZS*CMZ
          SL=XS*CLX+YS*CLY+ZS*CLZ
```

```
          AM=AX*CMX+AY*CMY+AZ*CMZ
          AL=AX*CLX+AY*CLY+AZ*CLZ
          BM=BX*CMX+BY*CMY+BZ*CMZ
          BL=BX*CLX+BY*CLY+BZ*CLZ
          PA=SM*(AL*AL+PNJK*PNJK)-AM*AL*SL
          PB=SM*(BL*BL+PNJK*PNJK)-BM*BL*SL
          RNUM=SL*PNJK*(A*PB-B*PA)
          DNOM=PA*PB+PNJK*PNJK*A*B*SL*SL

ctnm  the following is added to handle the special case of when the
c       point j lies in the plane of the panel k, as per VSAERO
c       theory document 4/8/93

          call cross(ax,ay,az,xs,ys,zs,rls,rly.rlz)
          rls=sqrt(rlx*rlx+rly*rly+rlz*rlz)
          rlsx=rlx/rls
          rlsy=rly/rls
          rlzz=rlz/rls

ctnm  take the dot product of the previous vector and the normal
c       vector of the panel to determine if the control point is on
c       the right or left side
c       if rlsn>0, rightside
c       if rlsn<0, leftside 4/8/93

          rlsn=rlsx*cnx+rlsy*cny+rlsz*cnz

ctnm  if the projected height approaches zero from the positive side

          if(pnjk.gt.0) then
            if(dnom.gt.0)then
               DUBINF=0.

ctnm if dnom "approximately" equals zero

            elseif(abs(dnom).lt.EPS)then

ctnm if on the "right" side

               if(rlsn.gt.0.)then
                  DUBINF=pi/2.

ctnm if on the "left"side
               else

                  DUBINF=-pi/2.
               endif

ctnm if dnom less than zero

            else
ctnm if on the "right" side

               if (rlsn.gt.0.)then
                  DUBINF=pi

ctnm if on the "left"side
               else

                  DUBINF=-pi
               endif
             endif

ctnm if the projected height approaches zero from the negative side
          else

            if(dnom.gt.0.)then

               DUBINF=0.

            elseif(abs(dnom).lt.EPS)then

ctnm if on the right side

               if(rlsn.gt.0.)then
                  DUBINF=-pi/2.
```

```
                 else

ctnm if on the left side
                    DUBINF=pi/2.
                 endif

ctnm if on the "left"side

                 else

ctnm if on the right side

                    if(rlsn.gt.0.)then
                      DUBINF=-pi

ctnm if on the left side

                    else

                      DUBINF=pi

                    endif

                  endif

                endif

              else

           DUBINF = ATAN2(RNUM,DNOM)

              endif
ctnm if working on the first wake column,(assumes wake is paneled from root
c       to tip), : use doublet effect based on scale= length of three sides
c          divided by the perimeter of the wake panel 4/15/93


           CJK1 = CJK1 + (scale*DUBINF)
           CJK = CJK + DUBINF

   10   CONTINUE
        RETURN
        END
```

Appendix E.

Subroutine LENGTH.f

```
      Subroutine Length(wptx,wpty,wptz,scale)
c-----------------------------------------------------------------------------
c
c  PURPOSE:    computes a ratio of the lengths of sides 1,2 & 3 to the peri-
c              meter of a wake panel (used to scale the first wake column
c              to be able to remove the wake panels from the body)
c              Assumes wake paneling is from root to tip
c
c  CALLED BY: wakinfl
c
c  EXTERNAL REFERENCES:  none
c
c  ENVIRONMENT:  VAX/VMS FORTRAN,CRAY CFT77 FORTRAN,
c                MACINTOSH DCM MACTRAN PLUS 3.0
c
c  AUTHOR:  Thomas N. Mouch,
c           KU FRL, University of Kansas, Lawrence, KS 66045
c
c  DEVELOPMENT HISTORY:
c       DATE   INITIALS  DESCRIPTION
c
c-----------------------------------------------------------------------------

      dimension wptx(5),wpty(5),wptz(5)
      sumnum=0.
      sumdnm = 0.

ctnm    step through the 4 sides

      do 10 npts=1,4
        dx=wptx(npts)-wptx(npts+1)
        dy=wpty(npts)-wpty(npts+1)
        dz=wptz(npts)-wptz(npts+1)
        side=sqrt(dx*dx + dy*dy + dz*dz)
        if (npts.le.3) sumnum=sumnum + side
        sumdnm = sumdnm + side
 10   continue

ctnm  determine the ratio of (sum of three sides)/perimeter

      scale = sumnum/sumdnm
      return
      end
```

Appendix F.

Subroutine RHS.f

```
*DECK RHS
      SUBROUTINE RHS
C
C-----------------------------------------------------------------------
C
C  PURPOSE:    THIS ROUTINE FORMS RIGHT-HAND SIDE VECTOR FOR SET OF EQUATIONS
C              BASED ON THE CURRENT FREE-STREAM CONDITIONS AND PRESET NORMAL
C              VELOCITIES
C
C  CALLED BY:  WAKINFL
C
C  EXTERNAL REFERENCES:   NONE
C
C  ENVIRONMENT:  VAX/VMS FORTRAN,CRAY CFT77 FORTRAN,
C                MACINTOSH DCM MACTRAN PLUS 3.0
C
C  AUTHOR:  Dale Ashby,
C           MS 247-2, NASA Ames Research Center, Moffett Field, CA.  94035
C
C  DEVELOPMENT HISTORY:
C       DATE   INITIALS  DESCRIPTION
C
C-----------------------------------------------------------------------
C
C
C  CODE DIMENSIONING PARAMETERS
C
C  NUMBER OF SURFACE PANELS ALLOWED
C
      PARAMETER (NSPDIM = 4000)
C
C  NUMBER OF NEUMANN PANELS ALLOWED
C
      PARAMETER (NNPDIM = 50)
C
C  NUMBER OF PATCHES ALLOWED
C
      PARAMETER (NPDIM = 20)
C
C  NUMBER OF BASIC POINTS ALLOWED FOR SECTION DEFINITION
C  (ALSO NUMBER OF SECTIONS ALLOWED PER PATCH)
C  (ALSO NUMBER OF ROWS OR COLUMNS + 1 ALLOWED ON A PATCH)
C  CAUTION:  DO NOT SET THIS PARAMETER TO LESS THAN 50)
C
      PARAMETER (NBPDIM = 100)
C
C  NUMBER OF WAKE PANELS ALLOWED
C
      PARAMETER (NWPDIM = 1500)
C
C  NUMBER OF WAKE COLUMNS ALLOWED ON EACH WAKE
C
      PARAMETER (NWCDIM = 50)
C
C  NUMBER OF WAKES ALLOWED
C
      PARAMETER (NWDIM = 50)
C
C  NUMBER OF SCAN VOLUMES OF EACH TYPE ALLOWED
C
      PARAMETER (NSVDIM = 10)
C
C  NUMBER OF POINTS PER OFF-BODY STREAMLINE ALLOWED
C
      PARAMETER (NSLPDIM = 1000)
C
C  NUMBER OF GROUPS OF PANELS ON WHICH NONZERO NORMAL VELOCITY IS PRESCRIBED
C
      PARAMETER (NVELDIM = 200)
C
```

```
C  NUMBER OF LINES AT A TIME TO BE READ IN FOR THE INFLUENCE COEF. MATRIX
C  IN THE SOLVER ROUTINE (BUFFERED INPUT FROM THE SCRATCH FILE)
C  (CAUTION:  DO NOT SET LARGER THAN ONE UNLESS YOU ARE SURE YOU HAVE
C  ENOUGH MEMORY TO HANDLE BUFFERED INPUT!)
C
      PARAMETER (MATBUF = 1)
C
C  NUMBER OF WAKE CORNER POINTS ALLOWED
C
      PARAMETER (NWCPDIM=(NWPDIM + 1)*2)
C
C  NUMBER OF SURFACE CORNER POINTS ALLOWED
C
      PARAMETER (NSCPDIM=(NSPDIM + 1)*2)
C
C  NUMBER OF EDGE PANELS ALLOWED ON A PATCH
C
      PARAMETER (NEPDIM = NEPDIM * 4)
C
ctnm  number of viscous data points to be read in 3/26/93
c
      parameter (nvpts = 30)
      DIMENSION  DUBIC(NSPDIM),
     +           VNP(NSPDIM),
     +           SORSIC(NSPDIM)
ctnm RLXFAC added 2/5/93
      COMMON/ CONST    / PI,      EPS,     FOURPI,  CBAR,
     +                   SSPAN,   SREF,    RMPX,    RMPY,     RMPZ,
     +                   MAXIT,    SOLRES,RLXFAC,
     +                   RCORS,   RFF
      COMMON/ SOLUTION  / SIG(NSPDIM), DUB(NSPDIM), PDUB(NSPDIM),
     +                    WDUB(NWPDIM), VX(NSPDIM), VY(NSPDIM),
     +                    VZ(NSPDIM), VXR(NNPDIM), VYR(NNPDIM),
     +                    VZR(NNPDIM),DIAG(NSPDIM),
     +                    RHSV(NSPDIM), VNORMAL(NSPDIM), CPDUB(NSCPDIM)
      COMMON/ PATCHES  / IDENT(NPDIM), IPAN(NPDIM), KLASS(NPDIM),
     +                   KOMP(NPDIM),  LPAN(NPDIM), NCOL(NPDIM),
     +                   NPANS(NPDIM), NROW(NPDIM)
      COMMON/ SPANEL  / XC(NSPDIM), YC(NSPDIM), ZC(NSPDIM),
     +                  PCS(3,3,NSPDIM), AREA(NSPDIM), PFF(NSPDIM),
     +                  CPSX(NSCPDIM), CPSY(NSCPDIM), CPSZ(NSCPDIM),
     +                  ICPS(NPDIM), KPTYP(NSPDIM),SMP(NSPDIM),
     +                  SMQ(NSPDIM)
      COMMON/ INTERNAL / NCZONE,NCZPAN,CZDUB ,VREF
      COMMON/ TSTEP    / NTSTPS, ITSTEP
      COMMON/ NUM     / NPAN,NPATCH,NWPAN,NWAKE,NCOMP,NASSEM
      COMMON/ ONSET    / ALPHA,ALDEG,YAW,YAWDEG,
     +                   BETA,WIND(3,3),PHIDOT,THEDOT,PSIDOT,
     +                   COMPOP,SYM,GPR,VINF,VSOUND
      COMMON/ PRINT    / LSTINP,   LSTOUT, LSTGEO, LSTNAB, LSTWAK,
     +                   LSTFRQ, LSTCPV ,LSTHLD, LSTJET
      COMMON/ SCRFILES / JPLOT, JDUBIC,
     +                   JSORIC, IMU
      COMMON/ UNSTDY   / OMEGA(3,10), VFR(3,10), DTSTEP
ctnm  VISCOUS added for iteration with viscous data 1/26/93
ctnm  dimensions increased to 30 3/26/93

      COMMON/ VISCOUS / IVISCS,IDENTV(NPDIM),IVPRNT,NVISC,
     +                  NPVMAX(10),ALP2D(10,nvpts),CL2D(10,nvpts),
     +                  CD2D(10,nvpts),CM2D(10,nvpts),ALPZRO(10),
     +                  rhsinc(NSPDIM)
ctnm added for viscous normal velocity 2/5/93
      real VFZnew(nspdim),vfxnew(nspdim)
      LOGICAL SYM,GPR
      REWIND JSORIC
      REWIND JDUBIC
C
C  SET UP THE CURRENT FREESTREAM VELOCITY VECTOR
C
      KMP = 1
      DO 10 NP=1,NPATCH
        VFX = VFR(1,KMP)
        VFY = VFR(2,KMP)
        VFZ = VFR(3,KMP)
        vcal=sqrt(vfx*vfx+vfz*vfz)
```

```
ctnm  add the increment in alpha to the current RHS vector 2/5/93

          do 7 j=ipan(np),lpan(np)
            VFZnew(j)=vinf*(-sin(asin(-VFZ/vinf) - rhsinc(j)))
            vfxnew(j)=vinf*(-cos(acos(-vfx/vinf) - rhsinc(j)))
 7        continue
        endif
        OMEGAX = OMEGA(1,KMP)
        OMEGAY = OMEGA(2,KMP)
        OMEGAZ = OMEGA(3,KMP)
C
C  FIND COMPONENT OF FREESTREAM VELOCITY NORMAL TO EACH PANEL AND ADD TO
C  ANY PRESET NORMAL VELOCITY FOR EACH PANEL
C
        DO 5 K=IPAN(NP),LPAN(NP)
          XN = PCS(1,3,K)
          YN = PCS(2,3,K)
          ZN = PCS(3,3,K)
          VRX = YC(K) * PCS(3,3,K) - ZC(K) * PCS(2,3,K)
          VRY = ZC(K) * PCS(1,3,K) - XC(K) * PCS(3,3,K)
          VRZ = XC(K) * PCS(2,3,K) - YC(K) * PCS(1,3,K)
ctnm  VFZ changed for viscous data 2/5/93
          if(ivprnt.eq.1)then
            VNP(K) = VNORMAL(K) + VFXnew(k) * XN + VFY * YN + VFZnew(k)
     +            * ZN + OMEGAX * VRX + OMEGAY * VRY + OMEGAZ * VRZ
          else
            VNP(K) = VNORMAL(K) + VFX * XN + VFY * YN + VFZ * ZN +
     +           OMEGAX * VRX + OMEGAY * VRY + OMEGAZ * VRZ
          endif
C
C  SET SOURCE VALUE FOR EACH PANEL
C
          IF(IDENT(NP).NE.3)THEN
            SIG(K) = VNP(K)/FOURPI
          ELSE
            SIG(K) = 0.0
          ENDIF
    5   CONTINUE
   10 CONTINUE
C
C  COMPUTE THE INITIAL RIGHT HAND SIDE VECTOR
C
      DO 20 NP=1,NPATCH
        DO 30 J=IPAN(NP),LPAN(NP)
          IF(NCZONE.GT.0)THEN
            READ(JDUBIC)(DUBIC(K),K=1,NPAN)
          ENDIF
          READ(JSORIC) (SORSIC(K),K=1,NPAN)
          EJ = 0.0
          DO 40 K=1,NPAN
            IF(NCZONE.GT.0.AND.K.EQ.NCZPAN)THEN
              EJ = EJ + CZDUB * DUBIC(K)
            ELSE
              EJ = EJ + SIG(K) * SORSIC(K)
            ENDIF
   40     CONTINUE
          RHSV(J) = EJ
          IF(IDENT(NP).EQ.3)RHSV(J) = EJ - VNP(J)
   30   CONTINUE
   20 CONTINUE
      REWIND JSORIC
      REWIND JDUBIC
  601 format(2x,i3,3x,6(2x,f10.5))
      RETURN
      END
```

Appendix G.

Subroutine SEARCH.F

```
*DECK SEARCH
      SUBROUTINE SEARCH(jmax,dalpi,cl,alpact,clcalc)
c
c-----------------------------------------------------------------------
c
c  PURPOSE:     searches through an ordered set of points for the point just
c               greater than the input value, performs a linear interpolation
c               between these two points and returns the interpolated value
c
c  CALLED BY:  VISCDATA
c
c  EXTERNAL REFERENCES:   NONE
c
c  ENVIRONMENT:  VAX/VMS FORTRAN,CRAY CFT77 FORTRAN,
c                MACINTOSH DCM MACTRAN PLUS 3.0
c
c  AUTHOR:  Thomas N. Mouch,
c           KU FRL, University of Kansas, Lawrence, KS 66045
c
c  DEVELOPMENT HISTORY:
c       DATE   INITIALS   DESCRIPTION
c
c-----------------------------------------------------------------------
c
c
c  CODE DIMENSIONING PARAMETERS
c
c  NUMBER OF SURFACE PANELS ALLOWED
c
      PARAMETER (NSPDIM = 4000)
c
c  NUMBER OF NEUMANN PANELS ALLOWED
c
      PARAMETER (NNPDIM = 50)
c
c  NUMBER OF PATCHES ALLOWED
c
      PARAMETER (NPDIM = 20)
c
c  NUMBER OF BASIC POINTS ALLOWED FOR SECTION DEFINITION
c  (ALSO NUMBER OF SECTIONS ALLOWED PER PATCH)
c  (ALSO NUMBER OF ROWS OR COLUMNS - 1 ALLOWED ON A PATCH)
c  CAUTION:  DO NOT SET THIS PARAMETER TO LESS THAN 50!
c
      PARAMETER (NBPDIM = 100)
c
c  NUMBER OF WAKE PANELS ALLOWED
c
      PARAMETER (NWPDIM = 1500)
c
c  NUMBER OF WAKE COLUMNS ALLOWED ON EACH WAKE
c
      PARAMETER (NWCDIM = 50)
c
c  NUMBER OF WAKES ALLOWED
c
      PARAMETER (NWDIM = 50)
c
c  NUMBER OF SCAN VOLUMES OF EACH TYPE ALLOWED
c
      PARAMETER (NSVDIM = 10)
c
c  NUMBER OF POINTS PER OFF-BODY STREAMLINE ALLOWED
c
      PARAMETER (NSLPDIM = 1000)
c
c  NUMBER OF GROUPS OF PANELS ON WHICH NONZERO NORMAL VELOCITY IS PRESCRIBED
c
      PARAMETER (NVELDIM = 200)
```

```
c
c  NUMBER OF LINES AT A TIME TO BE READ IN FOR THE INFLUENCE COEF. MATRIX
c  IN THE SOLVER ROUTINE (BUFFERED INPUT FROM THE SCRATCH FILE
c  (CAUTION:  DO NOT SET LARGER THAN ONE UNLESS YOU ARE SURE YOU HAVE
c  ENOUGH MEMORY TO HANDLE BUFFERED INPUT).
c
      PARAMETER (MATBUF = 1)
c
c  NUMBER OF WAKE CORNER POINTS ALLOWED
c
      PARAMETER (NWCPDIM=(NWPDIM + 1)*2)
c
c  NUMBER OF SURFACE CORNER POINTS ALLOWED
c
      PARAMETER (NSCPDIM=(NSPDIM + 1)*2)
c
c  NUMBER OF EDGE PANELS ALLOWED ON A PATCH
c
      PARAMETER (NEPDIM = NBPDIM * 4)
c
ctnm  number of viscous data points to be read in 3/26/93
c
      parameter (nvpts = 30)
        real dalpi(nvpts),alpact,cl(nvpts)

ctnm initialize the counter

        k=1

ctnm  set the flag showing a value hasn't been found

        iflag=0
 20     continue

ctnm  test to see if the current value is greater than the desired value

        if(dalpi(k).gt.alpact)then

ctnm  if it is greater, then if you have compared more than one value,
c                 interpolate

                if(k.gt.1) then
                        valu1=dalpi(k)
                        valu2=dalpi(k-1)
                        scale=(alpact-valu2)/(valu1-valu2)
                        clcalc=cl(k-1)+(scale*(cl(k)-cl(k-1)))

ctnm  a point has been found

                        iflag=1
                else

ctnm  if it isn't greater andonly one value compared,
ctnm   you're not in the data range,return the lowest value

                        write(9,*) 'out of range low, setting minimum'
                        clcalc=cl(1)

ctnm  a point has been found

                        iflag=1
                endif
        else

ctnm  if you have compared all of the values, then out of range high

                if(k.eq.jmax)then
                        write(9,*)'out of range high, setting max'
                        clcalc=cl(jmax)

ctnm  a point has been found

                        iflag=1
                endif
        endif
```

```
cthm if a point hasn't been found, increment the count and compare again

        if(iflag.eq.0)then
              k=k+1
              goto 20
        endif
        end
```

Appendix H.

Subroutine VISCDATA.f

```
*DECK viscdata
      SUBROUTINE viscdata
C
C-----------------------------------------------------------------------
C
C  PURPOSE:      EVALUATES THE CORRECTIONS REQUIRED TO THE SECTIONAL LIFT
C                COEFFICIENT TO INCLUDE VISCOUS EFFECTS IN THE INVISCID METHOD
C                CHANGES THE RHS VECTOR TO ACCOMODATE THESE EFFECTS
C
C  CALLED BY:  PMARC
C
C  EXTERNAL REFERENCES:   SEARCH,SOLVER,WAKDUE,NEUMANN,AERODAT
C
C  ENVIRONMENT:  VAX/VMS FORTRAN,CRAY CFT77 FORTRAN,
C                MACINTOSH DCM MACTRAN PLUS 3.0
C
C  AUTHOR:  Thomas N. Mouch,
C           KU FRL, University of Kansas, Lawrence, KS 66045
C
C  DEVELOPMENT HISTORY:
C       DATE   INITIALS   DESCRIPTION
C
C-----------------------------------------------------------------------
C
C
C  CODE DIMENSIONING PARAMETERS
C
C  NUMBER OF SURFACE PANELS ALLOWED
C
      PARAMETER (NSPDIM = 4000)
C
C  NUMBER OF NEUMANN PANELS ALLOWED
C
      PARAMETER (NNPDIM = 50)
C
C  NUMBER OF PATCHES ALLOWED
C
      PARAMETER (NPDIM = 20)
C
C  NUMBER OF BASIC POINTS ALLOWED FOR SECTION DEFINITION
C  (ALSO NUMBER OF SECTIONS ALLOWED PER PATCH)
C  (ALSO NUMBER OF ROWS OR COLUMNS + 1 ALLOWED ON A PATCH)
C  CAUTION:  DO NOT SET THIS PARAMETER TO LESS THAN 50.
C
      PARAMETER (NBPDIM = 100)
C
C  NUMBER OF WAKE PANELS ALLOWED
C
      PARAMETER (NWPDIM = 1500)
C
C  NUMBER OF WAKE COLUMNS ALLOWED ON EACH WAKE
C
      PARAMETER (NWCDIM = 50)
C
C  NUMBER OF WAKES ALLOWED
C
      PARAMETER (NWDIM = 50)
C
C  NUMBER OF SCAN VOLUMES OF EACH TYPE ALLOWED
C
      PARAMETER (NSVDIM = 10)
C
C  NUMBER OF POINTS PER OFF-BODY STREAMLINE ALLOWED
C
      PARAMETER (NSLPDIM = 1000)
C
C  NUMBER OF GROUPS OF PANELS ON WHICH NONZERO NORMAL VELOCITY IS PRESCRIBED
C
      PARAMETER (NVELDIM = 200)
C
```

```
c  NUMBER OF LINES AT A TIME TO BE READ IN FOR THE INFLUENCE COEF. MATRIX
c  IN THE SOLVER ROUTINE (BUFFERED INPUT FROM THE SCRATCH FILE)
c  (CAUTION:  DO NOT SET LARGER THAN ONE UNLESS YOU ARE SURE YOU HAVE
c  ENOUGH MEMORY TO HANDLE BUFFERED INPUT.)
c
      PARAMETER (MATBUF = 1)
c
c  NUMBER OF WAKE CORNER POINTS ALLOWED
c
      PARAMETER (NWCPDIM=(NWPDIM + 1)*2)
c
c  NUMBER OF SURFACE CORNER POINTS ALLOWED
c
      PARAMETER (NSCPDIM=(NSPDIM + 1)*2)
c
c  NUMBER OF EDGE PANELS ALLOWED ON A PATCH
c
      PARAMETER (NEPDIM = NBPDIM * 4)
c
ctnm  number of viscous data points to be read in 3/26/93
c
      parameter (nvpts = 30)
c
c  NUMBER OF VISCOUS ITERATIONS ALLOWED
c
      PARAMETER (itparm = 50)

ctnm RLXFAC added 2/5/93
      COMMON/ CONST    / PI,     EPS,     FOURPI,  CBAR,
     +                   SSPAN,  SREF,    RMPX,  RMPY,     RMPZ,
     +                   MAXIT,   SOLRES,RLXFAC,
     +                   RCORS,  RFF

ctnm added to iterate a solution with viscous data 1/15/93
ctnm updated to include section drag data 2/22/93
ctnm updated to include section moment and last pass info 3/19/93
ctnm updated to include section lift data 3/13/93
      common/ iterate /COLCLS(NPDIM,NBPDIM),COLCDS(NPDIM,NBPDIM),
     +                COLCMS(NPDIM,NBPDIM),TCLS,TCDS,tcms,
     +                dalpha(npdim,nbpdim),cd2dpt(npdim,nbpdim),
     +                cm2dpt(npdim,nbpdim),last,cl2dpt(npdim,nbpdim)

      COMMON/ NUM      / NPAN,NPATCH,NWPAN,NWAKE,NCOMP,NASSEM
      COMMON/ ONSET     / ALPHA,ALDEG,YAW,YAWDEG,
     +                   BETA,WIND(3,3),PHIDOT,THEDOT,PSIDOT,
     +                   COMPOP,SYM,GPR,VINF,VSOUND

      COMMON/ PATCHES  / IDENT(NPDIM), IPAN(NPDIM), KLASS(NPDIM),
     +                   KOMP(NPDIM),   LPAN(NPDIM), NCOL(NPDIM),
     +                   NPANS(NPDIM),  NROW(NPDIM)

      COMMON/ SOLUTION  / SIG(NSPDIM), DUB(NSPDIM), PDUB(NSPDIM),
     +                    WDUB(NWPDIM), VX(NSPDIM), VY(NSPDIM),
     +                    VZ(NSPDIM), VXR(NNPDIM), VYR(NNPDIM),
     +                    VZR(NNPDIM),DIAG(NSPDIM),
     +                    RHSV(NSPDIM), VNORMAL(NSPDIM), CPDUB(NSCPDIM)

ctnm  VISCOUS added for iteration with viscous data 1/19/93
ctnm  dimensions increased to 30 3/26/93

      COMMON/ VISCOUS / IVISCS,IDENTV(NPDIM),IVPRNT,NVISC,
     +                  NPVMAX(10),ALP2D(10,nvpts),CL2D(10,nvpts),
     +                  CD2D(10,nvpts),CM2D(10,nvpts),ALPZRO(10),
     +                  rhsinc(NSPDIM)

      real alxfer(nvpts),clxfer(nvpts),cdxfer(nvpts),cmxfer(nvpts)
      real cldiff(itparm),clstor(itparm),cdstor(itparm),
     +    cldifact(itparm),cmstor(itparm)
      radcon=180./pi
      rlxold = rlxfac

ctnm initialize a flag to show we have convergence and are making
ctnm    the last pass

      last = 0
      write(13,*)'alpha = ',alpha,'aldeg = ',aldeg
```

```
      if(compop.eq.1) then

ctnm if a compressible calculation, apply Prandtl-Glauert correction
          clslope=(2.* pi)/beta
      else
          clslope=2.* pi
      endif
      write (13,*)' RLXFAC = ',RLXFAC

ctnm initialize the count for number of cl iterations

      itcl=0
ctnm set a flag for whether to use a relaxation factor or not based
c      on is alphamax is exceeded 4/16/93
      irelax = 0

 5    continue

ctnm initialize the count for panel location
      k=1

ctnm count the number of iterations

      itcl=itcl+1

ctnm check to see if solution has converged

      if (last.eq.0) then

ctnm set a limit on number of iterations

        if(itcl.gt.itparm) then
          write(13,*) 'Maximum number of iterations exceeded, Cldiff = '
     +          ,cldifft
          last = 1
          goto 70
        endif
      endif

ctnm initialize delta alpha for first pass

      if (itcl.eq.1) then
          do 10 npit=1,npatch
                do 11 ncit=1,ncol(npit)
                      dalpha(npit,ncit)=0.
 11             continue
 10       continue
      endif
      do 20 np=1,npatch

ctnm added to try to "expedite" convergence on the tail 4/18/93
c  thus using a different relaxation factor on tail( pactch #2)

      if (np .eq. 2) then
        rlxfac = 2. * rlxfac
      else
        rlxfac = rlxold
      endif
ctnm if a wing patch, modify the circulation for the 2-d data

        if (ident(np).eq.1) then
          ivdata=identv(np)

ctnm  use the proper column of alpha and cl2d

            do 22 itran=1,npvmax(ivdata)
              alxfer(itran)=alp2d(ivdata,itran)
              clxfer(itran)=cl2d(ivdata,itran)
              cdxfer(itran)=cd2d(ivdata,itran)
              cmxfer(itran)=cm2d(ivdata,itran)
 22         continue
       write(13,*)'   k        alphae      Cl(3-d)      cl(2-d)          f
     +        delta a'
       write(13,*)'______    ________    _______     _______     _________
     +    _________'
```

```
          do 21 nc=1,ncol(np)
           clratio=colcls(np,nc)/clslope
           if(clratio.gt.1.0) then
              write(13,*) 'clratio > 1.0'
              clratio = 0.99
           endif

ctnm  compute alpha/geometric - alpha/induce  in degrees, due to lookup

           alpeff=(asin(clratio)+dalpha(np,nc) *radcon+alpzro(ivdata)
           if(alpeff.gt.aldeg)then
              write(13,*) 'alpeff > geometric alpha'
              alpeff=aldeg
           endif

           if(alpeff.gt.alxfer(npvmax(ivdata)) then
              alpeff=alxfer(npvmax(ivdata))
              irelax=1
           endif
           if(alpeff.lt.alxfer(1))alpeff=alxfer(1)

ctnm lookup the 2-d cl for this alpha

           call search(npvmax(ivdata),alxfer,clxfer,alpeff,
     +        cl2dpt(np,nc))

ctnm lookup the cd and cm only if this is the last pass

           if( last.eq.1) then
             call search(npvmax(ivdata),alxfer,cdxfer,alpeff,
     +          cd2dpt(np,nc))
             call search(npvmax(ivdata),alxfer,cmxfer,alpeff,
     +          cm2dpt(np,nc))
           endif
           ffactor=cl2dpt(np,nc)/colcls(np,nc)

ctnm added to mimic lan's program 1/25/93
           if(abs(colcls(np,nc)).lt.0.001)ffactor=1.
           if(ffactor.gt.1.4)ffactor=1.4
           if(ffactor.lt.-1.4)ffactor=-1.4
           fsina=ffactor*sin(alpha)
           if(abs(fsina).gt.1.)fsina=0.95*fsina/abs(fsina)
ctnm dalpha is in radians
           dalpold=dalpha(np,nc)
           dalpnew=alpha - asin(fsina)
           if(irelax.eq.0)then
             dalpha(np,nc)=(1-rlxfac)*dalpold + rlxfac*dalpnew
           else
             dalpha(np,nc)=dalpnew
             irelax=0
           endif
        write(13,600)nc,alpeff,colcls(np,nc)
     +     ,cl2dpt(np,nc),ffactor,dalpha(np,nc)*radcon
            do 30 nr=1,nrow(np)

ctnm  dalpha is "overrelaxed" overall by a factor of 1.1

                  rhsinc(k)=1.1*dalpha(np,nc)
                  k=k+1
 30         continue
 21        continue
        else

ctnm if not a wing patch, set the circulation increment to zero

          do 40 nc=1,ncol(np)
            do 50 nr=1,nrow(nc)
              rhsinc(k)=0.
              k=k+1
 50         continue
 40       continue
      endif
 20   continue

      call wakinfl
      call solver
```

```
      call wakdub
      call neumann
      clold=tcls
      cdold=tcds
      cmold=tcms
 70   call aerodat
      if(itcl.eq.1)then
        clstor(itcl)=clold
        cdstor(itcl)=cdold
        cmstor(itcl)=cmold
      endif
      cldiff(itcl+1)=abs(clold-tcls)/abs(clold
      cldifact(itcl+1)=(clold-tcls)/clold
ctnm next three variables are just for output
      clstor(itcl+1)=tcls
      cdstor(itcl+1)=tcds
      cmstor(itcl+1)=tcms
      cldifft=cldiff(itcl+1)
      write(13,*) 'itcl = ',itcl,'     % cldiff = ',cldifft*100.
      itmax=itcl
      if(last. eq. 1) goto 80
ctnm if change in CL less than 0.3% reduce rlxfac to prevent "chatter"
c     if(cldifft.lt.0.003)rlxfac=0.1*rlxfac

ctnm if change in CL less than 0.5%, do another iteration

      if (cldifft.gt.0.005) goto 5
      last = 1
      goto 5

 80   continue
      do 90 i=1,itmax-1
         write(13,602) i-1,cldifact(i),clstor(i),cdstor(i),cmstor(i)
 90   continue

ctnm reset the last flag when finished with viscous calculations 4/17/93

      last = 0

 600  format(i3,3x,5(2x,f9.5))
 601  format(2x,i3,5x,f8.3,2(2x,f8.3))
 602  format(2x,i3,5x,f8.4,3(2x,f8.4))
      end
```

Appendix I.

Subroutine WAKINFL.f

```
*DECK WAKINFL
      SUBROUTINE WAKINFL
C
C-----------------------------------------------------------------------------
C
C  PURPOSE:   CALCULATES THE WAKE POTENTIAL INFLUENCE AT THE SURFACE CONTROL
C             POINTS AND COMBINES THESE WITH THE SURFACE AND ONSET FLOW
C             IN ASSEMBLING THE COMPLETE MATRIX AT EACH STEP
C
C  CALLED BY:  PMARC
C
ctnm length added to scale the wake doublet influence for the first wake
c       column,  therefore don't need wakes from the body 4/15/93
C  EXTERNAL REFERENCES:   IDUBPOT,PTDUBPOT,RHS,DISTANCE,SCHEME,length
C
C  ENVIRONMENT:  VAX/VMS FORTRAN,CRAY CFT77 FORTRAN,
C                MACINTOSH DCM MACTRAN PLUS 3.0
C
C  AUTHOR:  Dale Ashby,
C           MS 247-2, NASA Ames Research Center, Moffett Field, CA.  94035
C
C  DEVELOPMENT HISTORY:
C       DATE   INITIALS  DESCRIPTION
C
C-----------------------------------------------------------------------------
C
C
C  CODE DIMENSIONING PARAMETERS
C
C  NUMBER OF SURFACE PANELS ALLOWED
C
      PARAMETER (NSPDIM = 4000)
C
C  NUMBER OF NEUMANN PANELS ALLOWED
C
      PARAMETER (NNPDIM = 50)
C
C  NUMBER OF PATCHES ALLOWED
C
      PARAMETER (NPDIM = 20)
C
C  NUMBER OF BASIC POINTS ALLOWED FOR SECTION DEFINITION
C  (ALSO NUMBER OF SECTIONS ALLOWED PER PATCH)
C  (ALSO NUMBER OF ROWS OR COLUMNS + 1 ALLOWED ON A PATCH)
C  CAUTION:  DO NOT SET THIS PARAMETER TO LESS THAN 50!
C
      PARAMETER (NBPDIM = 100)
C
C  NUMBER OF WAKE PANELS ALLOWED
C
      PARAMETER (NWPDIM = 1500)
C
C  NUMBER OF WAKE COLUMNS ALLOWED ON EACH WAKE
C
      PARAMETER (NWCDIM = 50)
C
C  NUMBER OF WAKES ALLOWED
C
      PARAMETER (NWDIM = 50)
C
C  NUMBER OF SCAN VOLUMES OF EACH TYPE ALLOWED
C
      PARAMETER (NSVDIM = 10)
C
C  NUMBER OF POINTS PER OFF-BODY STREAMLINE ALLOWED
C
      PARAMETER (NSLPDIM = 1000)
```

```
C
C  NUMBER OF GROUPS OF PANELS ON WHICH NONZERO NORMAL VELOCITY IS PRESCRIBED
C
      PARAMETER (NVELDIM = 200)
C
C  NUMBER OF LINES AT A TIME TO BE READ IN FOR THE INFLUENCE COEF. MATRIX
C  IN THE SOLVER ROUTINE (BUFFERED INPUT FROM THE SCRATCH FILE)
C  (CAUTION:  DO NOT SET LARGER THAN ONE UNLESS YOU ARE SURE YOU HAVE
C  ENOUGH MEMORY TO HANDLE BUFFERED INPUT.)
C
      PARAMETER (MATBUF = 1)
C
C  NUMBER OF WAKE CORNER POINTS ALLOWED
C
      PARAMETER (NWCPDIM=(NWPDIM + 1)*2)
C
C  NUMBER OF SURFACE CORNER POINTS ALLOWED
C
      PARAMETER (NSCPDIM=(NSPDIM + 1)*2)
C
C  NUMBER OF EDGE PANELS ALLOWED ON A PATCH
C
      PARAMETER (NEPDIM = NBPDIM * 4)
C
ctnm  number of viscous data points to be read in 3/26/93
c
      parameter (nvpts = 30)
      DIMENSION          SMPW(NWPDIM),SORSIC(NSPDIM),
     +                   DUBIC(NSPDIM,MATBUF),XSE(4),YSE(4)
      COMMON/ SOLUTION  / SIG(NSPDIM), DUB(NSPDIM), PDUB(NSPDIM),
     +                    WDUB(NWPDIM), VX(NSPDIM), VY(NSPDIM),
     +                    VZ(NSPDIM), VXR(NNPDIM), VYR(NNPDIM),
     +                    VZR(NNPDIM),DIAG(NSPDIM),
     +                    RHSV(NSPDIM), VNORMAL(NSPDIM), CPDUB(NSCPDIM)
      COMMON/ SPANEL  / XC(NSPDIM), YC(NSPDIM), ZC(NSPDIM),
     +                  PCS(3,3,NSPDIM), AREA(NSPDIM), PFF(NSPDIM),
     +                  CPSX(NSCPDIM), CPSY(NSCPDIM), CPSZ(NSCPDIM),
     +                  ICPS(NPDIM), KPTYP(NSPDIM),SMP(NSPDIM),
     +                  SMQ(NSPDIM)
ctnm RLXFAC added 2/5/93
      COMMON/ CONST    / PI,      EPS,     FOURPI,  CBAR,
     +                   SSPAN,   SREF,    RMPX,    RMPY,     RMPZ,
     +                   MAXIT,    SOLRES,RLXFAC,
     +                   RCORS,   RFF
      COMMON/ PATCHES  / IDENT(NPDIM), IPAN(NPDIM), KLASS(NPDIM),
     +                   KOMP(NPDIM),   LPAN(NPDIM), NCOL(NPDIM),
     +                   NPANS(NPDIM), NROW(NPDIM)
      COMMON/ TSTEP    / NTSTPS, ITSTEP
      COMMON/ INTERNAL / NCZONE,NCZPAN,CZDUB,VREF
      COMMON/ NUM      / NPAN,NPATCH,NWPAN,NWAKE,NCOMP,NASSEM
      COMMON/ ONSET    / ALPHA,ALDEG,YAW,YAWDEG,
     +                   BETA,WIND(3,3),PHIDOT,THEDOT,PSIDOT,
     +                   COMPOP,SYM,GPR,VINF,VSOUND
      COMMON/ WAKES / NWCOL(NWDIM), NWROW(NWDIM), IWPAN(NWDIM),
     +                LWPAN(NWDIM), IDENTW(NWDIM),
     +                KWPU(NWCDIM,NWDIM), KWPL(NWCDIM,NWDIM),
     +                PHIU(NWCDIM,NWDIM), PHIL(NWCDIM,NWDIM),
     +                IFLEXW(NWDIM)
      COMMON/ PRINT    / LSTINP,   LSTOUT, LSTGEO, LSTNAB, LSTWAK,
     +                   LSTFRQ, LSTCPV ,LSTHLD, LSTJET
      COMMON/ SCRFILES / JPLOT, JDUBIC,
     +                   JSORIC, IMU
      COMMON/ UNSTDY   / OMEGA(3,10), VFR(3,10), DTSTEP

ctnm  VISCOUS added for iteration with viscous data 1/28/93
ctnm  dimensions increased to 30 3/26/93

      COMMON/ VISCOUS / IVISCS,IDENTV(NPDIM),IVPRNT,NVISC,
     +                  NPVMAX(10),ALP2D(10,nvpts),CL2D(10,nvpts),
     +                  CD2D(10,nvpts),CM2D(10,nvpts),ALPZRO(10),
     +                  rhsinc(NSPDIM)

      COMMON/ WPANEL  / XCW(NWPDIM), YCW(NWPDIM), ZCW(NWPDIM),
     +                  PCSW(3,3,NWPDIM), AREAW(NWPDIM),
     +                  PFFW(NWPDIM),
     +                  CPWX(NWCPDIM), CPWY(NWCPDIM), CPWZ(NWCPDIM),
```

```
     -                    ICPW(NWDIM)
      DIMENSION SWX(5),SWY(5),SWZ(5),ICPWSUB(4)
ctnm  variables wpt? added to hold corner point information to determine
c       scale factors for first wake column to eliminate body wake 4/15/93
      dimension wptx(5),wpty(5), wptz(5)
      LOGICAL SYM,GPR,IPR,FAR

      CALL RHS

C  REWIND TAPES NEEDED FOR THIS SUBROUTINE
C
      REWIND JSORIC
      REWIND IMU
      REWIND JDUBIC
C
C  COMPUTE THE WAKE POTENTIAL INFLUENCE AT THE SURFACE CONTROL POINTS
C
      ICTR = 0
      KPCTR = 0
      IF(NPAN.GT.MATBUF)THEN
        NBUF = MATBUF
      ELSE
        NBUF = NPAN
      ENDIF
      DO 10 NP=1,NPATCH
        ID = IABS(IDENT(NP))
        DO 20 KP=IPAN(NP),LPAN(NP)
C
C  READ IN THE SURFACE POTENTIAL INFLUENCE COEFFICIENTS
C
          IF(NCZONE.GT.0)THEN
            READ(JSORIC) (SORSIC(K),K=1,NPAN)
          ENDIF
          ICTR = ICTR + 1
          IF(ICTR.GT.NBUF.OR.ICTR.EQ.1)THEN
            DO 25 I=1,NBUF
              READ(JDUBIC)(DUBIC(K,I),K=1,NPAN)
   25       CONTINUE
            ICTR = 1
          ENDIF
C
C  IF THERE ARE NO WAKES, SET UP THE RIGHT HAND SIDE VECTOR AND THE
C  DIAGONAL VECTOR.  MODIFY THE INFLUENCE COEFF. MATRIX FOR INTERNAL
C  FLOW MODELING IF REQUESTED IN INPUT DECK, THEN GO TO SOLVER.
C
          IF(NWAKE.LT.1)THEN
            RHSV(KP) = -RHSV(KP)
            DIAG(KP) = DUBIC(KP,ICTR)
            IF(NCZONE.GT.0)THEN
              DUBIC(NCZPAN,ICTR) = SORSIC(NCZPAN)
            ENDIF
            IF(ICTR.EQ.NBUF)THEN
              WRITE(IMU)((DUBIC(K,KK),K=1,NPAN),KK=1,NBUF)
              KPCTR = KPCTR + NBUF
              IF((NPAN-KPCTR).GE.MATBUF)THEN
                NBUF = MATBUF
              ELSE
                NBUF = NPAN - KPCTR
              ENDIF
            ENDIF
            GO TO 20
          ENDIF
          KWP = 0
          RHSSUM = 0
C
C  STEP THROUGH THE WAKES
C
          DO 30 NW=1,NWAKE
            IDW = IABS(IDENTW(NW))
C
C  COMPUTE THE INFLUENCE COEFFICIENTS OF EACH WAKE PANEL ON ALL THE
C  SURFACE PANELS.  SUM THESE INTO THE APPROPRIATE INFLUENCE COEFF.
C  FOR THE SURFACE SEPARATION PANELS.
C
            DO 40 NWC=1,NWCOL(NW)
              DO 50 NWR=1,NWROW(NW)
```

```
                 KWP = KWP + 1
ctnm  the following added to scale the doublet effect on the first
c         column of the wake.  This allows the wakes to be deleted from
c         the body 4/15/93
                 if((nwc.eq.1).and.(nwr.gt.1).and.(np.ne.nw))then
                   wptx(1) = cpwx(kwp)
                   wpty(1) = cpwy(kwp)
                   wptz(1) = cpwz(kwp)
                   wptx(2) = cpwx(kwp + nwrow(nw) + 1)
                   wpty(2) = cpwy(kwp + nwrow(nw) + 1)
                   wptz(2) = cpwz(kwp + nwrow(nw) + 1)
                   wptx(3) = cpwx(kwp + nwrow(nw) + 2)
                   wpty(3) = cpwy(kwp + nwrow(nw) + 2)
                   wptz(3) = cpwz(kwp + nwrow(nw) + 2)
                   wptx(4) = cpwx(kwp + 1)
                   wpty(4) = cpwy(kwp + 1)
                   wptz(4) = cpwz(kwp + 1)
                   wptx(5) = wptx(1)
                   wpty(5) = wpty(1)
                   wptz(5) = wptz(1)
                   call length(wptx,wpty,wptz,wscale)
c                  endif
                 else
                   wscale = 1.
                 endif
                 ISY = 1.0
                 IGP = 1.0
                 DUBICW = 0.0
ctnm dubicwl used to sum wake doublet effect to first column w/c body
c         wakes present
                 dubicwl = 0.0
  100            CONTINUE
                 XCP = XC(KP)
                 YCP = YC(KP) * ISY
c ground effect
                 if(igp.eq.-1) then
                  zcp=-2.*height-zc(kp)
                 else
                  ZCP = ZC(KP) * IGP
                 end if
                 CALL DISTANCE(PFFW(KWP),XCW(KWP),YCW(KWP),ZCW(KWP),
     +                         XCP,YCP,ZCP,FAR)
                 CNX = PCSW(1,3,KWP)
                 CNY = PCSW(2,3,KWP)
                 CNZ = PCSW(3,3,KWP)
                 IF(FAR)THEN
                   IF(IDENT(NP).NE.3)THEN
                     CALL PTDUBPOT(AREAW(KWP),XCW(KWP),YCW(KWP),ZCW(KWP),
     +                          CNX,CNY,CNZ,XCP,YCP,ZCP,CJK)
                   ELSE
                     PJKX = XCP - XCW(KWP)
                     PJKY = YCP - YCW(KWP)
                     PJKZ = ZCP - ZCW(KWP)
                     PN = PJKX * CNX + PJKY * CNY + PJKZ * CNZ
                     PJK = SQRT(PJKX**2 + PJKY**2 + PJKZ**2)
                     VJKX = AREAW(KWP)*(3*PN*PJKX-PJK**2*CNX)/PJK**5
                     VJKY = AREAW(KWP)*(3*PN*PJKY-PJK**2*CNY)/PJK**5
                     VJKZ = AREAW(KWP)*(3*PN*PJKZ-PJK**2*CNZ)/PJK**5
                     EJK = VJKX*PCS(1,3,KP) + VJKY*PCS(2,3,KP) +
     +                     VJKZ * PCS(3,3,KP)
                     CJK = EJK
                   ENDIF
                 ELSE
                   CALL SCHEME(NWROW(NW),NWCOL(NW),IWPAN(NW),KWP,
     +                         ICPW(NW),ICPWSUB)
                   DO 60 I=1,4
                     SWX(I) = CPWX(ICPWSUB(I))
                     SWY(I) = CPWY(ICPWSUB(I))
                     SWZ(I) = CPWZ(ICPWSUB(I))
   60              CONTINUE
                   IF(IDENT(NP).NE.3)THEN
ctnm  wscale added to pass scaling for doublet effect on first
c       wake column  4/15/93
                     CALL IDUBPOT(wscale,SWX,SWY,SWZ,XCW(KWP),YCW(KWP),
     +                         ZCW(KWP),CNX,CNY,CNZ,XCP,YCP,ZCP,CJK,cjkl)
                   ELSE
```

```
                    VJKX=0.0
                    VJKY=0.0
                    VJKZ=0.0
                    DO 70 II=1,4
                      NS = II
                      AX=XCP - SWX(NS)
                      AY=YCP - SWY(NS)
                      AZ=ZCP - SWZ(NS)
                      IF(NS.EQ.4)NS = 0
                      BX=XCP - SWX(NS+1)
                      BY=YCP - SWY(NS+1)
                      BZ=ZCP - SWZ(NS+1)
                      A=SQRT(AX*AX + AY*AY + AZ*AZ)
                      B=SQRT(BX*BX + BY*BY + BZ*BZ)
                      AVBX = BZ*AY - BY*AZ
                      AVBY = AZ*BX - BZ*AX
                      AVBZ = AX*BY - BX*AY
                      AVBS = AVBX*AVBX + AVBY*AVBY + AVBZ*AVBZ
                      IF(AVBS.LE.RCORS.OR.(A*A).LE.RCORS.OR.
     +                   (B*B).LE.RCORS)GO TO 70
                      ADB=AX*BX + AY*BY + AZ*BZ
                      SCALE=(A+B)/(A*B*(A*B + ADB))
                      VJKX=VJKX+SCALE*AVBX
                      VJKY=VJKY+SCALE*AVBY
                      VJKZ=VJKZ+SCALE*AVBZ
   70               CONTINUE
                    EJK = VJKX*PCS(1,3,KP) + VJKY*PCS(2,3,KP) +
     +                    VJKZ * PCS(3,3,KP)
                    CJK = EJK
                  ENDIF
                ENDIF
                DUBICW = DUBICW + CJK
ctnm  dubucwl is a summing variable to sum the effect of the wake doublets
c       for the first column 4/14/93
                if(nwc.eq.1) dubicw1 = dubicw1 + cjk1
                IF(SYM.AND.ISY.EQ.1)THEN
                  ISY = -1
                  GO TO 100
                ENDIF
                IF(GPR.AND.IGP.EQ.1)THEN
                  IGP = -1
                  ISY = 1
                  GO TO 100
                ENDIF
                IF(NWR.NE.1.AND.ITSTEP.GT.1)THEN
                  RHSSUM = RHSSUM + DUBICW * WDUB(KWP)
                ELSE
                  KU = KWPU(NWC,NW)
                  KL = KWPL(NWC,NW)
c                 DUBIC(KU,ICTR) = DUBIC(KU,ICTR) + DUBICW
c                 DUBIC(KL,ICTR) = DUBIC(KL,ICTR) - DUBICW
                   RHSSUM = RHSSUM + DUBICW*(PHIU(NWC,NW)-PHIL(NWC,NW))
ctnm added to account for wake changes 4/13/93
                  if((nwc.eq.1).and.(nwr.gt.1).and.(np.ne.nw))then
                    DUBIC(KU,ICTR) = DUBIC(KU,ICTR) + DUBICW1
                    DUBIC(KL,ICTR) = DUBIC(KL,ICTR) - DUBICW1
                  else
                    DUBIC(KU,ICTR) = DUBIC(KU,ICTR) + DUBICW
                    DUBIC(KL,ICTR) = DUBIC(KL,ICTR) - DUBICW
                  endif
                ENDIF
   50         CONTINUE
   40       CONTINUE
   30     CONTINUE
          RHSVT = RHSV(KP) + RHSSUM
          RHSV(KP) = -RHSVT
          DIAG(KP) = DUBIC(KP,ICTR)
          IF(NCZONE.GT.0)THEN
            DUBIC(NCZPAN,ICTR) = SORSIC(NCZPAN)
          ENDIF
          IF(ICTR.EQ.NBUF)THEN
            WRITE(IMU)((DUBIC(N,NN),N=1,NPAN),NN=1,NBUF)
            KPCTR = KPCTR + NBUF
            IF((NPAN-KPCTR).GE.MATBUF)THEN
              NBUF = MATBUF
            ELSE
```

```
            NBUF = NPAN - KPCTR
          ENDIF
        ENDIF
 20   CONTINUE
 10 CONTINUE
    REWIND IMU
    REWIND JSORIC
    RETURN
601 format(2x,i3,5x,f8.3,2x,f8.3,2x,f8.3)
    END
```

Appendix J.

Sample Input Dataset For T-tail Configuration

This dataset highlights the changes in input required to run a viscous case with different viscous datasets for the wing and tail. Additions to the input are in bold type. Input dataset is called "data5." (Large type is not part of the dataset!)

```
fwh alpha=2,t,wake def=+1,1/2cosine
&BINP2   LSTINP=0,     LSTOUT=1,     LSTFRQ=1,     LENRUN=0,                      &END
&BINP3   LSTGEO=0,     LSTNAB=0,     LSTWAK=0,     LSTCPV=0,     LSTJET=0,        &END
```

rlxfac is a relaxation factor. range: 0<rlxfac<2. Suggested value, rlxfac<0.5. Assuming the tail is patch #2, the value of rlxfac will be doubled to speed convergence for the tail

```
&BINP4   MAXIT=150,     SOLRES=0.0009,rlxfac=0.125   &END
&BINP5   NTSTPS=0,     DTSTEP=2.0,                                                &END
&BINP6   RSYM=0.0,     RGPR=0.0,     RFF=5.0,      RCORE=0.05,                    &END
```

iviscs sets whether viscous calculations are performed

default: inviscid case, iviscs=0
if viscous case, iviscs=1 (2-d data required for dataset "drag.")

Maximum of 10 viscous datasets is currently set.

```
&BINP7   VINF=1.0,     VSOUND=1116.0, UNIT=0,       COMPOP=0.0,iviscs=1   &END
&BINP8   ALDEG=2.0, YAWDEG=0.0, THEDOT=0.0, PSIDOT=0.0, PHIDOT=0.0,     &END
&BINP9   CBAR=.24606,   SREF=.48436,   SSPAN=.98424,
         RMPX=.86121,   RMPY=0.00,    RMPZ=0.00,                        &END
&BINP10  NORSET=0,     NBCHGE=0,     NCZONE=0,
         NCZPAN=0,     CZDUB=0.0,    VREF=00.0,                                   &END

&ASEM1   ASEMX=0.00,       ASEMY=0.00,       ASEMZ=0.00,
         ASCAL=1.00,       ATHET=0.00,       NODEA=5,                             &END

&COMP1   COMPX=    0.0000, COMPY=    0.0000, COMPZ=    0.0000,
         CSCAL=    1.0000, CTHET=   0.0,     NODEC= 5,                            &END
```

ivptch tells PMARC which viscous dataset (read in through "drag.") is to be applied to this patch. In "drag.", the angle of attack range is assumed to increase from a minimum to a maximum (and then start again for multiple datasets). A maximum of 30 angles of attack per dataset. Order of input across a row: α, c_l, c_d, c_m. The moment reference is assumed to be the quarterchord. If the reference is different the value of xmref must be changed in AERODAT.f. (This could all be changed to include the moment reference as part of the input in "drag." .) Here the main wing uses data from the first viscous data set.

```
&PATCH1  IREV=  0,IDPAT= 1,MAKE=   0,KCOMP= 1, KASS= 1,ivptch=1, &END
 main WING
&SECT1   STX= 0.73818,STY=    0.09022, STZ= 0.0000, SCALE= 0.24606,
         ALF=   0.0,  THETA=   0.0,
         INMODE=  5,  TNODS= 0,     TNPS=  0,     TINTS= 0,                       &END
&SECT2   RTC=    0.1200, RMC= 0.0400,    PPC= 0.4000,
         IPLANE=  2,  TNPC= 15,     TINTC= 0,                                     &END
&SECT1   STX= 0.73818, STY=    0.98424,STZ= 0.0000, SCALE= 0.24606,
         ALF=   0.0,  THETA=   0.0,
```

```
        INMODE=  0,  TNODS= 3,    TNPS= 20,    TINTS= 1,                      &END
```

The horizontal tail uses data from the second viscous dataset.

```
&PATCH1  IREV=  0, IDPAT= 1, MAKE=   0, KCOMP= 1, KASS= 1,ivptch=2, &END
 horizontal tail
&SECT1   STX=1.5912, STY= 0.0000,    STZ= 0.416670, SCALE= 0.16404,
         ALF=   0.0,  THETA=   0.0,
         INMODE=  5,  TNODS= 0,    TNPS=  0,    TINTS= 0,                 &END
&SECT2   RTC=    0.1200, RMC= 0.0000,    RPC= 0.0000,
         IPLANE=  2,  TNPC=10,    TINTC= 0,                               &END
&SECT1   STX= 1.5912, STY=0.36089,    STZ= 0.41667,    SCALE= .16404,
         ALF=   0.0,  THETA=   0.0,
         INMODE=  0,  TNODS= 3,    TNPS= 10,    TINTS= 2,                 &END

&PATCH1  IREV=  0,    IDPAT= 2,    MAKE=   0,   KCOMP= 1,    KASS= 1,     &END
NOSE
&SECT1   STX=  0.0000, STY=  0.0000, STZ=  0.0000, SCALE=  0.0000,
         ALF=   0.0,  THETA=   0.0,
         INMODE=  1,  TNODS= 0,    TNPS=  0,    TINTS= 0,   &END
   0.0000   -1.0000    0.0000
   0.1735   -0.9845    0.0000
   0.3420   -0.9395    0.0000
   0.5000   -0.8660    0.0000
   0.6425   -0.7660    0.0000
   0.7660   -0.6425    0.0000
   0.8660   -0.5000    0.0000
   0.9395   -0.3420    0.0000
   0.9845   -0.1735    0.0000
   1.0000    0.0000    0.0000
   0.9845    0.1735    0.0000
   0.9395    0.3420    0.0000
   0.8660    0.5000    0.0000
   0.7660    0.6425    0.0000
   0.6425    0.7660    0.0000
   0.5000    0.8660    0.0000
   0.3420    0.9395    0.0000
   0.1735    0.9845    0.0000
   0.0000    1.0000    0.0000
&BPNODE  TNODE= 3,    TNPC=  8,    TINTC=  3,               &END
&SECT1   STX=0.00049 , STY=    0.0000, STZ=    0.0000, SCALE= .00542,
         ALF=   0.0,  THETA=   0.0,
         INMODE=  0,  TNODS= 0,    TNPS=  0,    TINTS= 0,  &END
&SECT1   STX=0.00164 , STY=    0.0000, STZ=    0.0000, SCALE= .00989,
         ALF=   0.0,  THETA=   0.0,
         INMODE=  0,  TNODS= 0,    TNPS=  0,    TINTS= 0,  &END
&SECT1   STX=0.00328 , STY=    0.0000, STZ=    0.0000, SCALE= .01396,
         ALF=   0.0,  THETA=   0.0,
         INMODE=  0,  TNODS= 0,    TNPS=  0,    TINTS= 0,  &END
&SECT1   STX=0.00984 , STY=    0.0000, STZ=    0.0000, SCALE= .02404,
         ALF=   0.0,  THETA=   0.0,
         INMODE=  0,  TNODS= 0,    TNPS=  0,    TINTS= 0,  &END
&SECT1   STX=0.02625 , STY=    0.0000, STZ=    0.0000, SCALE= .03865,
         ALF=   0.0,  THETA=   0.0,
         INMODE=  0,  TNODS= 0,    TNPS=  0,    TINTS= 0,  &END
&SECT1   STX=0.04265 , STY=    0.0000, STZ=    0.0000, SCALE= .04648,
         ALF=   0.0,  THETA=   0.0,
         INMODE=  0,  TNODS= 0,    TNPS=  0,    TINTS= 0,  &END
&SECT1   STX=0.07546 , STY=    0.0000, STZ=    0.0000, SCALE= .06234,
         ALF=   0.0,  THETA=   0.0,
         INMODE=  0,  TNODS= 0,    TNPS=  0,    TINTS= 0,  &END
&SECT1   STX=0.10827 , STY=    0.0000, STZ=    0.0000, SCALE= .07201,
         ALF=   0.0,  THETA=   0.0,
         INMODE=  0,  TNODS= 0,    TNPS=  0,    TINTS= 0,  &END
&SECT1   STX=0.17388 , STY=    0.0000, STZ=    0.0000, SCALE= .08412,
         ALF=   0.0,  THETA=   0.0,
         INMODE=  0,  TNODS= 0,    TNPS=  0,    TINTS= 0,  &END
&SECT1   STX=0.27231 , STY=    0.0000, STZ=    0.0000, SCALE= .09022,
         ALF=   0.0,  THETA=   0.0,
         INMODE=  0,  TNODS= 2,    TNPS=  5,    TINTS= 0,  &END
&SECT1   STX=0.73818, STY=    0.0000, STZ=    0.0000, SCALE= .09022,
         ALF=   0.0,  THETA=   0.0,
         INMODE=  0,  TNODS= 3,    TNPS=  3,    TINTS= 0,  &END
```

```
&PATCH1  IREV=  0,    IDPAT= 2,    MAKE=   0,    KCOMP= 1,    KASS= 1,         &END
fuselage under wing
&SECT1   STX=0.73818, STY=    0.0000, STZ=    0.0000, SCALE= .09022,
         ALF=   0.0,  THETA=   0.0,
         INMODE=  1,  TNODS= 0,    TNPS=  0,    TINTS= 0,  &END
   0.0000   -1.0000    0.0000
   0.1735   -0.9845    0.0000
   0.3420   -0.9395    0.0000
   0.5000   -0.8660    0.0000
   0.6425   -0.7660    0.0000
   0.7660   -0.6425    0.0000
   0.8660   -0.5000    0.0000
   0.9395   -0.3420    0.0000
   0.9845   -0.1735    0.0000
   1.0000    0.0000    0.0000
&BPNODE  TNODE= 3,    TNPC=  4,     TINTC=  3,                   &END
&SECT1   STX=0.7399, STY=    0.0000, STZ=    0.0000, SCALE= .09022,
         ALF=   0.0,  THETA=   0.0,
         INMODE=  1,  TNODS= 0,    TNPS=  0,    TINTS= 0,  &END
   0.0000   -1.0000    0.0000
   0.1735   -0.9845    0.0000
   0.3420   -0.9395    0.0000
   0.5000   -0.8660    0.0000
   0.6425   -0.7660    0.0000
   0.7660   -0.6425    0.0000
   0.8660   -0.5000    0.0000
   0.9395   -0.3420    0.0000
   0.9845   -0.1735    0.0000
   1.0000   -0.0233    0.0000
&BPNODE  TNODE= 3,    TNPC=  4,     TINTC=  3,                   &END
&SECT1   STX=0.7470, STY=    0.0000, STZ=    0.0000, SCALE= .09022,
         ALF=   0.0,  THETA=   0.0,
         INMODE=  1,  TNODS= 0,    TNPS=  0,    TINTS= 0,  &END
   0.0000   -1.0000    0.0000
   0.1735   -0.9845    0.0000
   0.3420   -0.9395    0.0000
   0.5000   -0.8660    0.0000
   0.6425   -0.7660    0.0000
   0.7660   -0.6425    0.0000
   0.8660   -0.5000    0.0000
   0.9395   -0.3420    0.0000
   0.9845   -0.1735    0.0000
   1.0000   -0.0609    0.0000
&BPNODE  TNODE= 3,    TNPC=  4,     TINTC=  3,                   &END
&SECT1   STX=0.7599, STY=    0.0000, STZ=    0.0000, SCALE= .09022,
         ALF=   0.0,  THETA=   0.0,
         INMODE=  1,  TNODS= 0,    TNPS=  0,    TINTS= 0,  &END
   0.0000   -1.0000    0.0000
   0.1735   -0.9845    0.0000
   0.3420   -0.9395    0.0000
   0.5000   -0.8660    0.0000
   0.6425   -0.7660    0.0000
   0.7660   -0.6425    0.0000
   0.8660   -0.5000    0.0000
   0.9395   -0.3420    0.0000
   0.9845   -0.1735    0.0000
   1.0000   -0.0776    0.0000
&BPNODE  TNODE= 3,    TNPC=  4,     TINTC=  3,                   &END
&SECT1   STX=0.7773, STY=    0.0000, STZ=    0.0000, SCALE= .09022,
         ALF=   0.0,  THETA=   0.0,
         INMODE=  1,  TNODS= 0,    TNPS=  0,    TINTS= 0,  &END
   0.0000   -1.0000    0.0000
   0.1735   -0.9845    0.0000
   0.3420   -0.9395    0.0000
   0.5000   -0.8660    0.0000
   0.6425   -0.7660    0.0000
   0.7660   -0.6425    0.0000
   0.8660   -0.5000    0.0000
   0.9395   -0.3420    0.0000
   0.9845   -0.1735    0.0000
   1.0000   -0.0776    0.0000
&BPNODE  TNODE= 3,    TNPC=  4,     TINTC=  3,                   &END
&SECT1   STX=0.7983, STY=    0.0000, STZ=    0.0000, SCALE= .09022,
         ALF=   0.0,  THETA=   0.0,
         INMODE=  1,  TNODS= 0,    TNPS=  0,    TINTS= 0,  &END
   0.0000   -1.0000    0.0000
```

```
   0.1735   -0.9845    0.0000
   0.3420   -0.9395    0.0000
   0.5000   -0.8660    0.0000
   0.6425   -0.7660    0.0000
   0.7660   -0.6425    0.0000
   0.8660   -0.5000    0.0000
   0.9395   -0.3420    0.0000
   0.9845   -0.1735    0.0000
   1.0000   -0.0687    0.0000
&BPNODE  TNODE= 3,    TNPC=  4,     TINTC=  3,                &END
&SECT1   STX=0.8219, STY=    0.0000, STZ=    0.0000, SCALE= .09022,
         ALF=   0.0,  THETA=   0.0,
         INMODE=  1,  TNODS= 0,    TNPS=  0,     TINTS= 0,  &END
   0.0000   -1.0000    0.0000
   0.1735   -0.9845    0.0000
   0.3420   -0.9395    0.0000
   0.5000   -0.8660    0.0000
   0.6425   -0.7660    0.0000
   0.7660   -0.6425    0.0000
   0.8660   -0.5000    0.0000
   0.9395   -0.3420    0.0000
   0.9845   -0.1735    0.0000
   1.0000   -0.0565    0.0000
&BPNODE  TNODE= 3,    TNPC=  4,     TINTC=  3,                &END
&SECT1   STX=0.8473, STY=    0.0000, STZ=    0.0000, SCALE= .09022,
         ALF=   0.0,  THETA=   0.0,
         INMODE=  1,  TNODS= 0,    TNPS=  0,     TINTS= 0,  &END
   0.0000   -1.0000    0.0000
   0.1735   -0.9845    0.0000
   0.3420   -0.9395    0.0000
   0.5000   -0.8660    0.0000
   0.6425   -0.7660    0.0000
   0.7660   -0.6425    0.0000
   0.8660   -0.5000    0.0000
   0.9395   -0.3420    0.0000
   0.9845   -0.1735    0.0000
   1.0000   -0.0443    0.0000
&BPNODE  TNODE= 3,    TNPC=  4,     TINTC=  3,                &END
&SECT1   STX=0.8732, STY=    0.0000, STZ=    0.0000, SCALE= .09022,
         ALF=   0.0,  THETA=   0.0,
         INMODE=  1,  TNODS= 0,    TNPS=  0,     TINTS= 0,  &END
   0.0000   -1.0000    0.0000
   0.1735   -0.9845    0.0000
   0.3420   -0.9395    0.0000
   0.5000   -0.8660    0.0000
   0.6425   -0.7660    0.0000
   0.7660   -0.6425    0.0000
   0.8660   -0.5000    0.0000
   0.9395   -0.3420    0.0000
   0.9845   -0.1735    0.0000
   1.0000   -0.0321    0.0000
&BPNODE  TNODE= 3,    TNPC=  4,     TINTC=  3,                &END
&SECT1   STX=0.8986, STY=    0.0000, STZ=    0.0000, SCALE= .09022,
         ALF=   0.0,  THETA=   0.0,
         INMODE=  1,  TNODS= 0,    TNPS=  0,     TINTS= 0,  &END
   0.0000   -1.0000    0.0000
   0.1735   -0.9845    0.0000
   0.3420   -0.9395    0.0000
   0.5000   -0.8660    0.0000
   0.6425   -0.7660    0.0000
   0.7660   -0.6425    0.0000
   0.8660   -0.5000    0.0000
   0.9395   -0.3420    0.0000
   0.9845   -0.1735    0.0000
   1.0000   -0.0221    0.0000
&BPNODE  TNODE= 3,    TNPC=  4,     TINTC=  3,                &END
&SECT1   STX=0.9222, STY=    0.0000, STZ=    0.0000, SCALE= .09022,
         ALF=   0.0,  THETA=   0.0,
         INMODE=  1,  TNODS= 0,    TNPS=  0,     TINTS= 0,  &END
   0.0000   -1.0000    0.0000
   0.1735   -0.9845    0.0000
   0.3420   -0.9395    0.0000
   0.5000   -0.8660    0.0000
   0.6425   -0.7660    0.0000
   0.7660   -0.6425    0.0000
   0.8660   -0.5000    0.0000
```

```
   0.9395   -0.3420    0.0000
   0.9845   -0.1735    0.0000
   1.0000   -0.0144    0.0000
&BPNODE  TNODE= 3,    TNPC=  4,      TINTC=  3,               &END
&SECT1   STX=0.9432, STY=    0.0000, STZ=    0.0000, SCALE= .09022,
         ALF=   0.0,  THETA=   0.0,
         INMODE=  1,  TNODS= 0,    TNPS=  0,    TINTS= 0,  &END
   0.0000   -1.0000    0.0000
   0.1735   -0.9845    0.0000
   0.3420   -0.9395    0.0000
   0.5000   -0.8660    0.0000
   0.6425   -0.7660    0.0000
   0.7660   -0.6425    0.0000
   0.8660   -0.5000    0.0000
   0.9395   -0.3420    0.0000
   0.9845   -0.1735    0.0000
   1.0000   -0.0088    0.0000
&BPNODE  TNODE= 3,    TNPC=  4,      TINTC=  3,               &END
&SECT1   STX=0.9606, STY=    0.0000, STZ=    0.0000, SCALE= .09022,
         ALF=   0.0,  THETA=   0.0,
         INMODE=  1,  TNODS= 0,    TNPS=  0,    TINTS= 0,  &END
   0.0000   -1.0000    0.0000
   0.1735   -0.9845    0.0000
   0.3420   -0.9395    0.0000
   0.5000   -0.8660    0.0000
   0.6425   -0.7660    0.0000
   0.7660   -0.6425    0.0000
   0.8660   -0.5000    0.0000
   0.9395   -0.3420    0.0000
   0.9845   -0.1735    0.0000
   1.0000   -0.0055    0.0000
&BPNODE  TNODE= 3,    TNPC=  4,      TINTC=  3,               &END
&SECT1   STX=0.9735, STY=    0.0000, STZ=    0.0000, SCALE= .09022,
         ALF=   0.0,  THETA=   0.0,
         INMODE=  1,  TNODS= 0,    TNPS=  0,    TINTS= 0,  &END
   0.0000   -1.0000    0.0000
   0.1735   -0.9845    0.0000
   0.3420   -0.9395    0.0000
   0.5000   -0.8660    0.0000
   0.6425   -0.7660    0.0000
   0.7660   -0.6425    0.0000
   0.8660   -0.5000    0.0000
   0.9395   -0.3420    0.0000
   0.9845   -0.1735    0.0000
   1.0000   -0.0044    0.0000
&BPNODE  TNODE= 3,    TNPC=  4,      TINTC=  3,               &END
&SECT1   STX=0.9814, STY=    0.0000, STZ=    0.0000, SCALE= .09022,
         ALF=   0.0,  THETA=   0.0,
         INMODE=  1,  TNODS= 0,    TNPS=  0,    TINTS= 0,  &END
   0.0000   -1.0000    0.0000
   0.1735   -0.9845    0.0000
   0.3420   -0.9395    0.0000
   0.5000   -0.8660    0.0000
   0.6425   -0.7660    0.0000
   0.7660   -0.6425    0.0000
   0.8660   -0.5000    0.0000
   0.9395   -0.3420    0.0000
   0.9845   -0.1735    0.0000
   1.0000   -0.0044    0.0000
&BPNODE  TNODE= 3,    TNPC=  4,      TINTC=  3,               &END
&SECT1   STX=0.9842, STY=    0.0000, STZ=    0.0000, SCALE= .09022,
         ALF=   0.0,  THETA=   0.0,
         INMODE=  1,  TNODS= 3,    TNPS=  15,    TINTS= 0,  &END
   0.0000   -1.0000    0.0000
   0.1735   -0.9845    0.0000
   0.3420   -0.9395    0.0000
   0.5000   -0.8660    0.0000
   0.6425   -0.7660    0.0000
   0.7660   -0.6425    0.0000
   0.8660   -0.5000    0.0000
   0.9395   -0.3420    0.0000
   0.9845   -0.1735    0.0000
   1.0000    0.0000    0.0000
&BPNODE  TNODE= 3,    TNPC=  4,      TINTC=  3,               &END

&PATCH1  IREV=  0,    IDPAT= 2,    MAKE=   0,   KCOMP= 1,    KASS= 1,       &END
```

```
fuselage over wing
&SECT1   STX=0.7382, STY=    0.0000, STZ=    0.0000, SCALE= .09022,
         ALF=   0.0, THETA=   0.0,
         INMODE= 1,  TNODS= 0,    TNPS=  0,    TINTS= 0,  &END
   1.0000    0.0000    0.0000
   0.9845    0.1735    0.0000
   0.9395    0.3420    0.0000
   0.8660    0.5000    0.0000
   0.7660    0.6425    0.0000
   0.6425    0.7660    0.0000
   0.5000    0.8660    0.0000
   0.3420    0.9395    0.0000
   0.1735    0.9845    0.0000
   0.0000    1.0000    0.0000
&BPNODE  TNODE= 3,   TNPC=  4,    TINTC=  3,              &END
&SECT1   STX=0.7383, STY=    0.0000, STZ=    0.0000, SCALE= .09022,
         ALF=   0.0, THETA=   0.0,
         INMODE= 1,  TNODS= 0,    TNPS=  0,    TINTS= 0,  &END
   1.0000    0.0310    0.0000
   0.9845    0.1735    0.0000
   0.9395    0.3420    0.0000
   0.8660    0.5000    0.0000
   0.7660    0.6425    0.0000
   0.6425    0.7660    0.0000
   0.5000    0.8660    0.0000
   0.3420    0.9395    0.0000
   0.1735    0.9845    0.0000
   0.0000    1.0000    0.0000
&BPNODE  TNODE= 3,   TNPC=  4,    TINTC=  3,              &END
&SECT1   STX=0.7444, STY=    0.0000, STZ=    0.0000, SCALE= .09022,
         ALF=   0.0, THETA=   0.0,
         INMODE= 1,  TNODS= 0,    TNPS=  0,    TINTS= 0,  &END
   1.0000    0.0920    0.0000
   0.9845    0.1735    0.0000
   0.9395    0.3420    0.0000
   0.8660    0.5000    0.0000
   0.7660    0.6425    0.0000
   0.6425    0.7660    0.0000
   0.5000    0.8660    0.0000
   0.3420    0.9395    0.0000
   0.1735    0.9845    0.0000
   0.0000    1.0000    0.0000
&BPNODE  TNODE= 3,   TNPC=  4,    TINTC=  3,              &END
&SECT1   STX=0.7565, STY=    0.0000, STZ=    0.0000, SCALE= .09022,
         ALF=   0.0, THETA=   0.0,
         INMODE= 1,  TNODS= 0,    TNPS=  0,    TINTS= 0,  &END
   1.0000    0.1563    0.0000
   0.9845    0.1735    0.0000
   0.9395    0.3420    0.0000
   0.8660    0.5000    0.0000
   0.7660    0.6425    0.0000
   0.6425    0.7660    0.0000
   0.5000    0.8660    0.0000
   0.3420    0.9395    0.0000
   0.1735    0.9845    0.0000
   0.0000    1.0000    0.0000
&BPNODE  TNODE= 3,   TNPC=  4,    TINTC=  3,              &END
&SECT1   STX=0.7736, STY=    0.0000, STZ=    0.0000, SCALE= .09022,
         ALF=   0.0, THETA=   0.0,
         INMODE= 1,  TNODS= 0,    TNPS=  0,    TINTS= 0,  &END
   1.0000    0.2118    0.0000
   0.9845    0.2118    0.0000
   0.9395    0.3420    0.0000
   0.8660    0.5000    0.0000
   0.7660    0.6425    0.0000
   0.6425    0.7660    0.0000
   0.5000    0.8660    0.0000
   0.3420    0.9395    0.0000
   0.1735    0.9845    0.0000
   0.0000    1.0000    0.0000
&BPNODE  TNODE= 3,   TNPC=  4,    TINTC=  3,              &END
&SECT1   STX=0.7949, STY=    0.0000, STZ=    0.0000, SCALE= .09022,
         ALF=   0.0, THETA=   0.0,
         INMODE= 1,  TNODS= 0,    TNPS=  0,    TINTS= 0,  &END
   1.0000    0.2506    0.0000
   0.9845    0.2506    0.0000
```

```
  0.9395    0.3420    0.0000
  0.8660    0.5000    0.0000
  0.7660    0.6425    0.0000
  0.6425    0.7660    0.0000
  0.5000    0.8660    0.0000
  0.3420    0.9395    0.0000
  0.1735    0.9845    0.0000
  0.0000    1.0000    0.0000
&BPNODE  TNODE= 3,     TNPC=  4,     TINTC=  3.                &END
&SECT1   STX=0.8193, STY=    0.0000, STZ=    0.0000, SCALE= .09022,
         ALF=   0.0,  THETA=   0.0,
         INMODE=  1,  TNODS= 0,    TNPS=  0,     TINTS= 0,  &END
  1.0000    0.2694    0.0000
  0.9845    0.2694    0.0000
  0.9395    0.3420    0.0000
  0.8660    0.5000    0.0000
  0.7660    0.6425    0.0000
  0.6425    0.7660    0.0000
  0.5000    0.8660    0.0000
  0.3420    0.9395    0.0000
  0.1735    0.9845    0.0000
  0.0000    1.0000    0.0000
&BPNODE  TNODE= 3,     TNPC=  4,     TINTC=  3,                &END
&SECT1   STX=0.8454, STY=    0.0000, STZ=    0.0000, SCALE= .09022,
         ALF=   0.0,  THETA=   0.0,
         INMODE=  1,  TNODS= 0,    TNPS=  0,     TINTS= 0,  &END
  1.0000    0.2627    0.0000
  0.9845    0.2627    0.0000
  0.9395    0.3420    0.0000
  0.8660    0.5000    0.0000
  0.7660    0.6425    0.0000
  0.6425    0.7660    0.0000
  0.5000    0.8660    0.0000
  0.3420    0.9395    0.0000
  0.1735    0.9845    0.0000
  0.0000    1.0000    0.0000
&BPNODE  TNODE= 3,     TNPC=  4,     TINTC=  3,                &END
&SECT1   STX=0.8720, STY=    0.0000, STZ=    0.0000, SCALE= .09022,
         ALF=   0.0,  THETA=   0.0,
         INMODE=  1,  TNODS= 0,    TNPS=  0,     TINTS= 0,  &END
  1.0000    0.2395    0.0000
  0.9645    0.2395    0.0000
  0.9395    0.3420    0.0000
  0.8660    0.5000    0.0000
  0.7660    0.6425    0.0000
  0.6425    0.7660    0.0000
  0.5000    0.8660    0.0000
  0.3420    0.9395    0.0000
  0.1735    0.9845    0.0000
  0.0000    1.0000    0.0000
&BPNODE  TNODE= 3,     TNPC=  4,     TINTC=  3,                &END
&SECT1   STX=0.8980, STY=    0.0000, STZ=    0.0000, SCALE= .09022,
         ALF=   0.0,  THETA=   0.0,
         INMODE=  1,  TNODS= 0,    TNPS=  0,     TINTS= 0,  &END
  1.0000    0.2039    0.0000
  0.9845    0.2039    0.0000
  0.9395    0.3420    0.0000
  0.8660    0.5000    0.0000
  0.7660    0.6425    0.0000
  0.6425    0.7660    0.0000
  0.5000    0.8660    0.0000
  0.3420    0.9395    0.0000
  0.1735    0.9845    0.0000
  0.0000    1.0000    0.0000
&BPNODE  TNODE= 3,     TNPC=  4,     TINTC=  3,                &END
&SECT1   STX=0.9221, STY=    0.0000, STZ=    0.0000, SCALE= .09022,
         ALF=   0.0,  THETA=   0.0,
         INMODE=  1,  TNODS= 0,    TNPS=  0,     TINTS= 0,  &END
  1.0000    0.1608    0.0000
  0.9845    0.1735    0.0000
  0.9395    0.3420    0.0000
  0.8660    0.5000    0.0000
  0.7660    0.6425    0.0000
  0.6425    0.7660    0.0000
  0.5000    0.8660    0.0000
  0.3420    0.9395    0.0000
```

```
   0.1735    0.9845    0.0000
   0.0000    1.0000    0.0000
&BPNODE  TNODE= 3,     TNPC=  4,     TINTC=  3,               &END
&SECT1   STX=0.9433, STY=   0.0000, STZ=    0.0000, SCALE= .09022,
         ALF=   0.0,  THETA=   0.0,
         INMODE=  1,  TNODS= 0,    TNPS=  0,    TINTS= 0,  &END
   1.0000    0.1142    0.0000
   0.9845    0.1735    0.0000
   0.9395    0.3420    0.0000
   0.8660    0.5000    0.0000
   0.7660    0.6425    0.0000
   0.6425    0.7660    0.0000
   0.5000    0.8660    0.0000
   0.3420    0.9395    0.0000
   0.1735    0.9845    0.0000
   0.0000    1.0000    0.0000
&BPNODE  TNODE= 3,     TNPC=  4,     TINTC=  3,               &END
&SECT1   STX=0.9607, STY=   0.0000, STZ=    0.0000, SCALE= .09022,
         ALF=   0.0,  THETA=   0.0,
         INMODE=  1,  TNODS= 0,    TNPS=  0,    TINTS= 0,  &END
   1.0000    0.0709    0.0000
   0.9845    0.1735    0.0000
   0.9395    0.3420    0.0000
   0.8660    0.5000    0.0000
   0.7660    0.6425    0.0000
   0.6425    0.7660    0.0000
   0.5000    0.8660    0.0000
   0.3420    0.9395    0.0000
   0.1735    0.9845    0.0000
   0.0000    1.0000    0.0000
&BPNODE  TNODE= 3,     TNPC=  4,     TINTC=  3,               &END
&SECT1   STX=0.9737, STY=   0.0000, STZ=    0.0000, SCALE= .09022,
         ALF=   0.0,  THETA=   0.0,
         INMODE=  1,  TNODS= 0,    TNPS=  0,    TINTS= 0,  &END
   1.0000    0.0355    0.0000
   0.9845    0.1735    0.0000
   0.9395    0.3420    0.0000
   0.8660    0.5000    0.0000
   0.7660    0.6425    0.0000
   0.6425    0.7660    0.0000
   0.5000    0.8660    0.0000
   0.3420    0.9395    0.0000
   0.1735    0.9845    0.0000
   0.0000    1.0000    0.0000
&BPNODE  TNODE= 3,     TNPC=  4,     TINTC=  3,               &END
&SECT1   STX=0.9815, STY=   0.0000, STZ=    0.0000, SCALE= .09022,
         ALF=   0.0,  THETA=   0.0,
         INMODE=  1,  TNODS= 0,    TNPS=  0,    TINTS= 0,  &END
   1.0000    0.0122    0.0000
   0.9845    0.1735    0.0000
   0.9395    0.3420    0.0000
   0.8660    0.5000    0.0000
   0.7660    0.6425    0.0000
   0.6425    0.7660    0.0000
   0.5000    0.8660    0.0000
   0.3420    0.9395    0.0000
   0.1735    0.9845    0.0000
   0.0000    1.0000    0.0000
&BPNODE  TNODE= 3,     TNPC=  4,     TINTC=  3,               &END
&SECT1   STX=0.9842, STY=   0.0000, STZ=    0.0000, SCALE= .09022,
         ALF=   0.0,  THETA=   0.0,
         INMODE=  1,  TNODS= 3,    TNPS=  15,    TINTS= 0,  &END
   1.0000    0.0000    0.0000
   0.9845    0.1735    0.0000
   0.9395    0.3420    0.0000
   0.8660    0.5000    0.0000
   0.7660    0.6425    0.0000
   0.6425    0.7660    0.0000
   0.5000    0.8660    0.0000
   0.3420    0.9395    0.0000
   0.1735    0.9845    0.0000
   0.0000    1.0000    0.0000
&BPNODE  TNODE= 3,     TNPC=  4,     TINTC=  3,               &END
&PATCH1  IREV=  0,     IDPAT= 2,     MAKE=   0,   KCOMP= 1,     KASS= 1,       &END
center fuselage
&SECT1   STX=  0.9842, STY=  0.0000, STZ=  0.0000, SCALE=  0.09022,
```

```
        ALF=   0.0,  THETA=   0.0,
        INMODE=  1,  TNODS= 0,    TNPS=  0,    TINTS= 0,   &END
  0.0000   -1.0000   0.0000
  0.1735   -0.9845   0.0000
  0.3420   -0.9395   0.0000
  0.5000   -0.8660   0.0000
  0.6425   -0.7660   0.0000
  0.7660   -0.6425   0.0000
  0.8660   -0.5000   0.0000
  0.9395   -0.3420   0.0000
  0.9845   -0.1735   0.0000
  1.0000    0.0000   0.0000
  0.9845    0.1735   0.0000
  0.9395    0.3420   0.0000
  0.8660    0.5000   0.0000
  0.7660    0.6425   0.0000
  0.6425    0.7660   0.0000
  0.5000    0.8660   0.0000
  0.3420    0.9395   0.0000
  0.1735    0.9845   0.0000
  0.0000    1.0000   0.0000
&BPNODE TNODE= 3,    TNPC=  8,    TINTC=  3,              &END
&SECT1    STX=  1.5912, STY=  0.0000, STZ=  0.0000, SCALE=  0.09022,
        ALF=   0.0,  THETA=   0.0,
        INMODE=  0,  TNODS= 3,    TNPS=  5,    TINTS= 3,   &END

&PATCH1 IREV=  0,    IDPAT= 2,    MAKE=   0,   KCOMP= 1,    KASS= 1,      &END
fuselage under ttail
&SECT1    STX=1.5912, STY=    0.0000, STZ=    0.0000, SCALE= .09022,
        ALF=   0.0,  THETA=   0.0,
        INMODE=  1,  TNODS= 0,    TNPS=  0,    TINTS= 0,  &END
  0.0000   -1.0000   0.0000
  0.1735   -0.9845   0.0000
  0.3420   -0.9395   0.0000
  0.5000   -0.8660   0.0000
  0.6425   -0.7660   0.0000
  0.7660   -0.6425   0.0000
  0.8660   -0.5000   0.0000
  0.9395   -0.3420   0.0000
  0.9845   -0.1735   0.0000
  1.0000    0.0000   0.0000
  0.9845    0.1735   0.0000
  0.9395    0.3420   0.0000
  0.8660    0.5000   0.0000
  0.7660    0.6425   0.0000
  0.6425    0.7660   0.0000
  0.5000    0.8660   0.0000
  0.3420    0.9395   0.0000
  0.1735    0.9845   0.0000
  0.0000    1.0000   0.0000
&BPNODE TNODE= 3,    TNPC=  8,    TINTC=  3,              &END
&SECT1    STX=1.7552, STY  =  0.0000, STZ=  0.0000, SCALE=  0.09022,
        ALF=   0.0,  THETA=   0.0,
        INMODE=  0,  TNODS= 3,    TNPS= 10,    TINTS= 0,   &END

&PATCH1 IREV=  0,    IDPAT= 2,    MAKE=   0,   KCOMP= 1,    KASS= 1,      &END
aft fuselage
&SECT1    STX=1.7552, STY  =  0.0000, STZ=  0.0000, SCALE=  0.09022,
        ALF=   0.0,  THETA=   0.0,
        INMODE=  1,  TNODS= 0,    TNPS=  0,    TINTS= 0,   &END
  0.0000   -1.0000   0.0000
  0.1735   -0.9845   0.0000
  0.3420   -0.9395   0.0000
  0.5000   -0.8660   0.0000
  0.6425   -0.7660   0.0000
  0.7660   -0.6425   0.0000
  0.8660   -0.5000   0.0000
  0.9395   -0.3420   0.0000
  0.9845   -0.1735   0.0000
  1.0000    0.0000   0.0000
  0.9845    0.1735   0.0000
  0.9395    0.3420   0.0000
  0.8660    0.5000   0.0000
  0.7660    0.6425   0.0000
  0.6425    0.7660   0.0000
  0.5000    0.8660   0.0000
```

```
  0.3420     0.9395     0.0000
  0.1725     0.9845     0.0000
  0.0000     1.0000     0.0000
&BPNODE  TNODE= 3,    TNPC=  8,     TINTC=  3.              &END
&SECT1   STX=  1.834, STY=  0.0000, STZ=  0.0000, SCALE=  0.09022,
         ALF=   0.0,  THETA=   0.0,
         INMODE=  0,  TNODS= 5,     TNPS=  3,     TINTS= 3,   &END
```

NOTE: Wakes only need separate from lifting surfaces. Input direction is assumed from root to tip

```
&WAKE1   IDWAK=1,         IFLXW=0,                                          &END
 WING wake
&WAKE2   KWPACH=1,        KWSIDE=2,        KWLINE=0,        KWPAN1=0,
         KWPAN2=0,        NODEW=3,         INITIAL=1,                       &END
&SECT1   STX=   50.0, STY= 0.0000, STZ= 0.7418, SCALE=    1.0000,
         ALF=   0.0,  THETA=   0.0,
         INMODE=-1,   TNODS= 3,     TNPS= 20,     TINTS= 1,                 &END

&WAKE1   IDWAK=1,         IFLXW=0,                                          &END
htail wake
&WAKE2   KWPACH=2,        KWSIDE=2,        KWLINE=0,        KWPAN1=0,
         KWPAN2=0,        NODEW=5,         INITIAL=1,                       &END
&SECT1   STX=  50.0, STY= 0.0000, STZ= 0.7418, SCALE=    1.0000,
         ALF=   0.0,  THETA=   0.0,
         INMODE=-1,   TNODS= 3,     TNPS= 20,     TINTS= 1,                 &END

&VS1     NVOLR= 0,        NVOLC= 0,                                         &END

&SLIN1   NSTLIN=0,                                                          &END
```

Appendix K.

Sample Output From Data4.

This output dataset shows the typical output from data4. for a viscous calculation. This one is specifically the output for the example input dataset of Appendix J. "alpha zero" gives the calculated α_o for each dataset read in through "drag."

```
alpha zero = -4.000000000
alpha zero = 0.0000000000E+00
```

This line shows the number of wake iteration step, α, TCLS, TCDS, TCMS.

```
                                1.0000    2.0000    .5068    .0152    .0164
alpha =  0.3 90658104E-01 aldeg =  2.000000000
 RLXFAC =  0.1250000000
```

K is the column number across the lifting surface.

k	alphae	Cl (3-d)	cl (2-d)	f	delta a
1	.92779	.53973	.43614	.80807	.04800
2	1.04003	.55199	.44549	.80707	.04825
3	1.06907	.55516	.44808	.80712	.04823
4	1.05264	.55337	.44654	.80696	.04827
5	1.00862	.54856	.44288	.80735	.04818
6	.94277	.54136	.43739	.80794	.04803
7	.85431	.53170	.43002	.80876	.04782
8	.74369	.51961	.42080	.80983	.04756
9	.60842	.50483	.40952	.81122	.04721
10	.44503	.48696	.39591	.81301	.04676
11	.24939	.46557	.37960	.81535	.04618
12	.01749	.44021	.36028	.81843	.04541
13	-.25553	.41034	.34179	.83294	.04178
14	-.57297	.37559	.32062	.85365	.03660
15	-.93784	.33564	.29630	.88278	.02931
16	-1.34664	.29087	.26905	.92498	.01876
17	-1.78577	.24276	.23977	.98771	.00307
18	-2.22526	.19459	.20218	1.03900	-.00975
19	-2.65660	.14731	.14017	.95156	.01211
20	-3.14048	.09425	.07520	.79790	.05054

k	alphae	Cl (3-d)	cl (2-d)	f	delta a
1	.73772	.08090	.07575	.93634	.03184
2	.71269	.07815	.07318	.93634	.03184
3	.66707	.07315	.06849	.93633	.03184
4	.60648	.06651	.06227	.93633	.03185
5	.53614	.05879	.05505	.93633	.03185
6	.45948	.05039	.04718	.93632	.03185
7	.37516	.04114	.03852	.93632	.03185
8	.26362	.02891	.02707	.93632	.03185
9	-.01299	-.00142	-.00104	.72951	.13528
10	-.36693	-.04024	-.02935	.72952	.13528

```
                                1.0000    2.0000    .5010    .0150    .0169
itcl =        1         % cldiff =   1.15663716569542885
```

k	alphae	Cl (3-d)	cl (2-d)	f	delta a
1	.93623	.53540	.43684	.81591	.08803
2	1.04704	.54748	.44608	.81478	.08854
3	1.07528	.55057	.44871	.81498	.08847
4	1.05867	.54875	.44705	.81466	.08859
5	1.01465	.54395	.44338	.81510	.08839
6	.94921	.53682	.43792	.81578	.08810
7	.86139	.52725	.43061	.81671	.08768
8	.75157	.51527	.42145	.81792	.08714

k	alphae	Cl(3-d)	cl(2-d)	f	delta a
9	.61742	.50065	.41027	.81949	.08645
10	.45546	.48299	.39678	.82150	.08555
11	.26150	.46184	.38061	.82411	.08439
12	.03173	.43680	.36146	.82753	.08286
13	-.24141	.40731	.34273	.84144	.07621
14	-.56026	.37298	.32147	.86191	.06656
15	-.92814	.33350	.29695	.89041	.05306
16	-1.34348	.28916	.26926	.93118	.03363
17	-1.79456	.24146	.23919	.99060	.00504
18	-2.24403	.19360	.19948	1.03037	-.01613
19	-2.65138	.14655	.14092	.96159	.02021
20	-3.09453	.09375	.07923	.84507	.08297

k	alphae	Cl(3-d)	cl(2-d)	f	delta a
1	.74828	.07856	.07683	.97795	.03491
2	.72394	.07590	.07433	.97942	.03418
3	.67888	.07095	.06971	.98242	.03268
4	.61925	.06441	.06358	.98709	.03034
5	.55043	.05687	.05652	.99383	.02697
6	.47527	.04863	.04880	1.00358	.02210
7	.39266	.03957	.04032	1.01898	.01440
8	.28207	.02744	.02896	1.05551	-.00388
9	.11707	-.00200	.01202	-1.40000	1.30160
10	-.20572	-.03739	-.01646	.44011	.38144

1.0000 2.0000 .4977 .0148 .0161

itcl = 2 % cldif = 0.657950947061181068

k	alphae	Cl(3-d)	cl(2-d)	f	delta a
1	.94332	.53180	.43743	.82255	.12141
2	1.05290	.54372	.44657	.82131	.12215
3	1.08035	.54673	.44921	.82164	.12202
4	1.06355	.54488	.44753	.82134	.12219
5	1.01955	.54009	.44379	.82168	.12194
6	.95446	.53302	.43836	.82242	.12149
7	.86726	.52353	.43109	.82344	.12088
8	.75833	.51169	.42202	.82476	.12008
9	.62523	.49721	.41093	.82646	.11904
10	.46455	.47974	.39753	.82864	.11771
11	.27204	.45882	.38149	.83146	.11599
12	.04388	.43403	.36248	.83515	.11373
13	-.22943	.40485	.34353	.84852	.10456
14	-.54962	.37086	.32218	.86873	.09107
15	-.92057	.33173	.29745	.89668	.07226
16	-1.34137	.28776	.26940	.93619	.04538
17	-1.80233	.24039	.23867	.99284	.00620
18	-2.25763	.19281	.19753	1.02446	-.02023
19	-2.64865	.14596	.14131	.96815	.02564
20	-3.06568	.09336	.08175	.87566	.10369

k	alphae	Cl(3-d)	cl(2-d)	f	delta a
1	.75158	.07859	.07717	.98195	.03521
2	.72643	.07591	.07459	.98257	.03435
3	.68001	.07099	.06982	.98360	.03271
4	.61756	.06439	.06341	.98471	.03040
5	.54395	.05669	.05585	.98518	.02764
6	.46086	.04812	.04732	.98348	.02484
7	.36406	.03834	.03738	.97487	.02337
8	.20747	.02318	.02130	.91913	.03754
9	1.25676	-.00492	.12532	-1.40000	2.17633
10	.30890	-.00795	.03172	-1.40000	1.48621

1.0000 2.0000 .4948 .0147 .0161

itcl = 3 % cldiff = 0.58296043425798416l

k	alphae	Cl(3-d)	cl(2-d)	f	delta a
1	.94956	.52884	.43795	.82814	.14921
2	1.05798	.54060	.44699	.82683	.15019
3	1.08472	.54354	.44965	.82726	.14996
4	1.06804	.54170	.44798	.82699	.15018
5	1.02411	.53693	.44417	.82724	.14990
6	.95919	.52988	.43876	.82803	.14931
7	.87232	.52046	.43152	.82911	.14850
8	.76393	.50870	.42248	[illegible]	.14745
9	.63165	.49435	.41146	.83232	.14609
10	.47185	.47703	.39814	.83464	.14435
11	.28048	.45629	.38219	.83762	.14210
12	.05355	.43171	.36328	.84150	.13915

13	-.21996	.40279	.34416	.85444	.12789
14	-.54127	.36909	.32274	.87441	.11109
15	-.91465	.33027	.29785	.90183	.08778
16	-1.34015	.28661	.26948	.94024	.05466
17	-1.80922	.23951	.23821	.99458	.00678
18	-2.26780	.19215	.19607	1.02040	-.02280
19	-2.64779	.14546	.14144	.97235	.02935
20	-3.04806	.09302	.08390	.90195	.11525

k	alphae	Cl(3-d)	cl(2-d)	f	delta a
1	.75153	.07855	.07717	.98237	.03523
2	.72595	.07584	.07454	.98285	.03435
3	.67848	.07082	.06967	.98376	.03266
4	.61449	.06405	.06309	.98507	.03027
5	.53866	.05604	.05531	.98698	.02725
6	.45296	.04695	.04651	.99064	.02331
7	.35481	.03635	.03643	1.00233	.01636
8	.20634	.01851	.02119	1.14454	-.04415

Here the $\alpha_e < \alpha_{geometric}$, therefore α_e is set equal to $\alpha_{geometric}$.

alpeff > geometric alpha

9	2.00000	-.01200	.19304	-1.40000	2.83239
10	1.47725	-.00098	.14541	1.00000	1.11466

1.0000 2.0000 .4926 .0146 .0152

itcl = 4 % cldiff = 0.434468220919370651

k	alphae	Cl(3-d)	cl(2-d)	f	delta a
1	.95480	.52637	.43839	.83285	.17236
2	1.06239	.53802	.44742	.83159	.17353
3	1.08861	.54091	.45004	.83200	.17323
4	1.07180	.53905	.44836	.83175	.17349
5	1.02800	.53430	.44449	.83192	.17320
6	.96332	.52729	.43910	.83274	.17248
7	.87677	.51792	.43189	.83388	.17148
8	.76889	.50625	.42290	.83536	.17019
9	.63709	.49199	.41191	.83724	.16854
10	.47803	.47479	.39866	.83965	.16641
11	.28749	.45420	.38278	.84276	.16366
12	.06147	.42979	.36394	.84678	.16007
13	-.21215	.40109	.34468	.85936	.14708
14	-.53446	.36765	.32319	.87908	.12745
15	-.90992	.32909	.29816	.90602	.10031
16	-1.33942	.28567	.26953	.94349	.06196
17	-1.81509	.23880	.23782	.99589	.00696
18	-2.27540	.19159	.19497	1.01763	-.02436
19	-2.64790	.14504	.14142	.97505	.03193
20	-3.03903	.09274	.08519	.91864	.12119

k	alphae	Cl(3-d)	cl(2-d)	f	delta a
1	.75364	.07878	.07738	.98225	.03530
2	.72808	.07608	.07476	.98269	.03442
3	.68027	.07102	.06985	.98355	.03272
4	.61593	.06422	.06324	.98473	.03034
5	.53914	.05613	.05536	.98616	.02736
6	.45136	.04694	.04634	.98731	.02383
7	.34599	.03615	.03553	.98279	.02088
8	.11195	.01712	.01150	.67150	.13118

alpeff > geometric alpha

9	2.00000	-.01103	.19304	-1.40000	3.32443
10	1.32699	.02328	.13172	1.40000	.63586

This is a listing of the values used in calculating C_D and C_M .

ncol	xle	xdist	zle	zdist	arearat	cl2dpt	cm2dpt
1	.7382	.0615	.0000	.0000	.0365	.4384	-.0898
2	.7382	.0615	.0000	.0000	.0362	.4474	-.0894
3	.7382	.0615	.0000	.0000	.0358	.4500	-.0893
4	.7382	.0615	.0000	.0000	.0351	.4484	-.0894
5	.7382	.0615	.0000	.0000	.0342	.4445	-.0895

6	.7382	.0615	.0000	.0000	.0331	.4391	-.0897	
7	.7382	.0615	.0000	.0000	.0318	.4319	-.0900	
8	.7382	.0615	.0000	.0000	.0303	.4229	-.0903	
9	.7382	.0615	.0000	.0000	.0287	.4119	-.0908	
10	.7382	.0615	.0000	.0000	.0268	.3987	-.0913	
11	.7382	.0615	.0000	.0000	.0248	.3828	-.0919	
12	.7382	.0615	.0000	.0000	.0226	.3639	-.0926	
13	.7382	.0615	.0000	.0000	.0203	.3447	-.0934	
14	.7382	.0615	.0000	.0000	.0178	.3232	-.0942	
15	.7382	.0615	.0000	.0000	.0153	.2982	-.0952	
16	.7382	.0615	.0000	.0000	.0126	.2695	-.0964	
17	.7382	.0615	.0000	.0000	.0099	.2378	-.0977	
18	.7382	.0615	.0000	.0000	.0071	.1950	-.0985	
19	.7382	.0615	.0000	.0000	.0043	.1414	-.0985	
20	.7382	.0615	.0000	.0000	.0014	.0852	-.0985	
1	1.5912	-.7710	.4167	.4167	.0195	.0774	-.0002	
2	1.5912	-.7710	.4167	.4167	.0190	.0748	-.0002	
3	1.5912	-.7710	.4167	.4167	.0180	.0698	-.0002	
4	1.5912	-.7710	.4167	.4167	.0167	.0632	-.0003	
5	1.5912	-.7710	.4167	.4167	.0149	.0554	-.0003	
6	1.5912	-.7710	.4167	.4167	.0127	.0463	-.0004	
7	1.5912	-.7710	.4167	.4167	.0102	.0355	-.0005	
8	1.5912	-.7710	.4167	.4167	.0075	.0115	-.0007	
9	1.5912	-.7710	.4167	.4167	.0046	.1930	.0039	
10	1.5912	-.7710	.4167	.4167	.0015	.1317	.0014	
				1.0000	2.0000	.4907	.0288	.0057

itcl = 5 & cldiff = 0.391655042767524719

This is a listing of the iteration sequence. Values given are: viscous iteratin number (0 is the initial inviscid calculation.), $C_{L\ diff}$, C_L, C_D, C_M.

0	.0000	.5068	.0152	.0164
1	.0116	.5010	.0150	.0169
2	.0066	.4977	.0148	.0161
3	.0058	.4948	.0147	.0161
4	.0043	.4926	.0146	.0152
5	.0039	.4907	.0288	.0057